A Ubiquitous Approach for Next Generation Information Systems

Tarek H. El-Basuny

Department of Information and Computer Science,
King Fahd University of Petroleum and Minerals,
KFUPM # 37, Dhahran 31261, Saudi Arabia, Mail Box 413,
Tel: 966-3-860-1967 & Fax: 966-3-860-2174,
Email: helmy@ccse.kfupm.edu.sa

Abstract

Recently, the spread of mobile technologies and communication infrastructures has made the vision of ubiquitous computing much more realistic and feasible. At the same time, agent technologies have attracted a lot of interest in both academe and industry as an emerging programming paradigm. We present the system, which is being developed as a multi-agent-based approach that lets the users ubiquitously retrieve more relevant information from the distributed Web portals. In particular, we developed agent-based framework, where each agent is autonomous, articulate, and social. *We reported methods to embed the autonomous portal agents into the Web portals, to cluster the portal agents into communities, to exploit and adapt the semantic policies by the Web mining agent and the attributes of the Web portal by the portal agent adaptively. In order to investigate the performance of the system, we carried out several experiments and developed a smart query routing mechanism for routing the user's query. Through the experiments, the results ensure that the proposed system promises to achieve more relevant information to the user's queries.*

Keywords: Parallel and distributed systems, Agent self-organization, Clustering, Routing, Learning and Adaptation.

1. Introduction

Recently, agent technologies have attracted a lot of interest in both academe and industry as an emerging programming paradigm. With such agent technologies, services are created by collaboration among agents. At the same time, the spread of mobile technologies and communication infrastructures have made it possible to access the network anytime and from anywhere. Ubiquitous computing plus ubiquitous sensing, facilitates search in ways only yet to be appreciated expand my experience with personalized Web search [13, 14, 15, 29]. Web search is a natural and everyday aspect of human activity, where we are looking for relevant information to satisfy a perceived need of some information.

The problem is that finding the information that an individual desires is often quite difficult, because of the complexity in organization and the quantity of information stored. The models behind current search-engines only access the static content of the Web, while the Web is however highly dynamic [22, 10]. Making this immense amount of information available for ubiquitous computing in daily life is a great challenge. In this environment, to realize ubiquitous systems, we propose a new agent-oriented information system that provides the user with relevant information and is completely different from the current search engines populated on the Internet. Researchers in the Artificial Intelligence (AI) and the Information Retrieval (IR) fields have already succeeded in developing multi-agent based techniques to automate tedious tasks and to facilitate the management of information flooding [8, 9, 11, 17, 20, 21, 26]. In this paper we will start by describing the agent system hierarchy, introduce the novel methodologies of clustering the system's agents into communities, the routing mechanism of the system and the evaluation on the adaptability of the system's agents. Finally we present the experimental results and future work of the system.

2. Agent System Hierarchy

The basic idea behind the work we describe in this paper is to embed intelligent agents called portal agents into the distributed Web portals as a way to partially address the problems posted by the dynamic nature of the Web. These portal agents can manifest various levels of intelligent behavior from simply reactive to adaptive and learning, where agents actually learn what users like and dislike. The implemented system uses four basic components [Fig. 1], which are: Interface Agent (IA), Router Agent (RA), Portal Agent (PA) and Page Mining Agent (PMA). The IA assigned to each user's machine and is usually a running process that operates in parallel with the user, communicates with the PAs via a RA. The RA delegates the user's query to the most popular and relevant PA to the query. The PA is designed as a special server extension module [27], which learns to function in social environments and where necessary collaborates, completes or negotiates with other agents. The PA creates a PMA for each Web page in the Web portal based on the preexisting Web links infrastructure [9, 19], and is responsible for starting the search process and running of the registered PMAs. Once started, the PMAs, which can be seen as the local representatives of the Web pages, can access the local data of the Web pages and create the *Semantic Polices* (*SP*). At the retrieval phase, the PMA uses the *SP* to decide whether or not the user's query belongs to the PMA. There is a clear mapping between the problem of searching in our system and the classic AI searching paradigm. Each PMA is a node and the hypertext links to other down chain PMAs are the edges of the graph to be searched. In typical AI domains a good heuristic will rate nodes higher as we progress towards some goal node. In the system, the heuristic model means how a page is relevant to the given query. The PA and the PMAs while interacting with the known or down chain agents use a standard best-first search algorithm. It has been slightly modified so that it will finish after reaching a predefined depth value, and returns the best PAs or PMAs, which have relevant information to the user's query.

We present an architecture aimed to support the semantic Web application, because from the application point of view, agents allow many of the functionalities the semantic Web promises, agents are accessing, manipulating, integrating Web content from heterogeneous resources and making inferences about the relationships among linked Web pages. However, if the Web pages have been semantically annotated and the agents could gather semantic tags from the Web pages, then the agents would know better to search these Web pages and their links. Moreover, the PA can use the gathered semantic information to refine the Web-crawling process.

Top

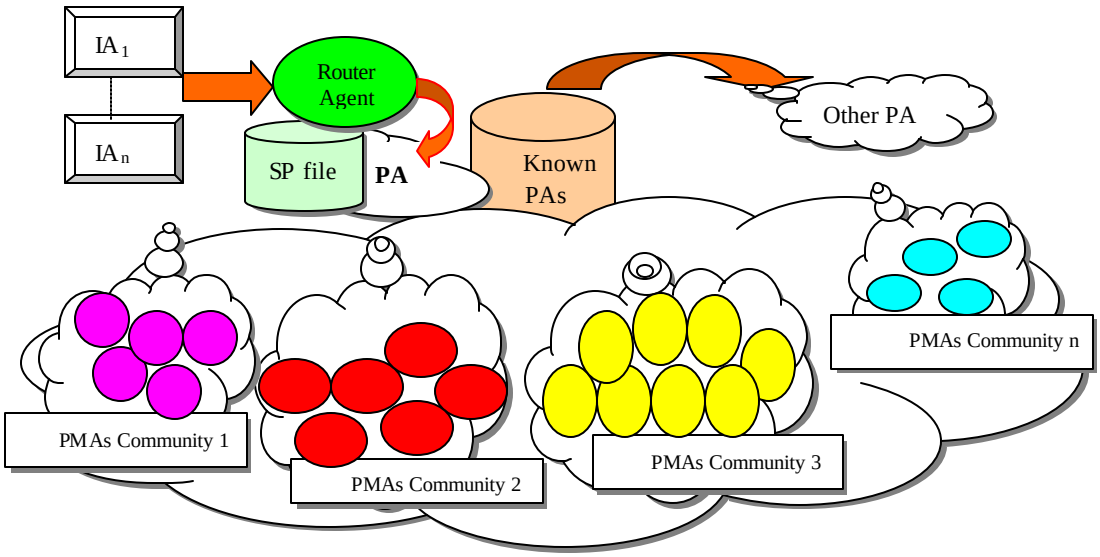

Figure 1 The hierarchical structure of system's agents

2.1 The Interface Agent

The IA resides in a user's machine and is usually a running process that operates in parallel with the user, communicates with the PAs via a RA to retrieve information relevant to the user's query. The IA is designed to learn and maintain the User's Preferences (UP) either explicitly or implicitly from his/her browsing behavior. The IA shows the results returned by the PMAs to the user after filtering and re-ranking them [12, 14]. The IA receives user's responses of his/her interest/not interest to the results and regards them as rewards to the results. For the IA to be truly useful UP must be inferred implicitly from actions and not obtained exclusively from explicit content ratings provided by the user, because having to stop to enter explicit ratings can alter normal patterns of browsing and reading. A more intelligent method is to use implicit ratings, where a rating is obtained by a method other than obtaining it directly from the user. By observing the browsing behavior of the user, it is possible to infer his/her implicit feedback without requiring the explicit judgments. Previous studies have shown that reading time to be a useful source of predicting UP implicitly [1, 2, 4, 5, 6, 7, 18, 24]. We investigated other sensors in correlation with the elapsed time of visiting the page to make the IA detects the actual user's implicit response. We developed the IA's browser to record the user's implicit ratings and the explicit rating of a Web page [14, 15]. Followings are the IA's job stream:

- The user starts by submitting a Natural Language (NL) query to the IA.
- The IA analyzes the NL query using a simple NL algorithm, throws out non-relevant words and transforms it to Q_{in}, where $Q_{in} = \langle k_1, k_2, ..., k_n \rangle$ stands for a vector of the keywords of the query.
- The IA looks for relevant URLs to the Q_{in} in the UP files.
- If the IA finds relevant URLs in the UP files, then it shows them to the user and asks the user whether he/she is satisfied or wants to search the Web for other sites.
- If the IA could not find in its UP files any URLs relevant to Q_{in} then the IA submits the query to the RA that routes Q_{in} to a relevant PA, which in turn forwards Q_{in} to its PMAs.
- The IA receives the results returned by the PA via the RA. The results consist of a set of Web pages and their similarity value to the given query.

Top

- The IA takes a set of queries from the UP files, whose similarity to Q_{in} is over the predefined threshold value, and creates a vector from the set of queries and Q_{in} in order to be used for filtering the retrieved results.
- The user explicitly marks the relevant pages using IA's feedback menu or the IA implicitly catches user's response. The response is used to adapt the content of UP files.

2.2 The Portal Agent

A PA is assigned to one Web portal to be responsible. The PA creates the hyper structure of the PMAs communities starting from the portal address of the Web portal. The PA knows all the PMAs in the Web server and works as a gateway when the PMAs communicate with each other or with one in another PA. The PA initiates all the PMAs in its domain when it starts searching for relevant information to the user's query. The PA clusters the PMAs into communities and automatically defines its attributes to be used in the routing mechanism. We introduce a definition of the PMAs community that enables the PA to effectively focus on narrow but topically related subsets of PMAs and to increase the precision of the search results. A PA of n Web pages creates a SP-file. In the SP-file, each SP denoted by $SP_i (1 \le i \le n)$, where n is the number of PMAs, is represented as vectors of keywords sorted in alphabetical order, $SP_i = \langle T_{ij} W_{ij} \mid 1 \le j \le t \rangle$, where $T_{ij} W_{ij}$ are a keyword and its weight, and t is the number of keywords in the SP_i. The weight value of the keyword decided by the frequency of the keyword and the kind of HTML tags that include the keyword in the Web page and is modified according to the user's responses. While creating a PMA for each Web page, the PA adds to its known PAs table all the portal addresses of the external links, which exist in the Web pages and point to other Web portals. This means that, the PAs community will be created automatically. The PMAs send "friend of mine" messages to the PA to register the portal addresses of the external links in their Web pages as known PAs.

2.2.1 Portal Agents Community

The PA will add to its address book all the portal addresses of the external links. This means that, the PAs community will be created automatically. For instances, while embedding the PA to the Web server of KFUPM http://www.kfup.ed.sa, if there are external links, i.e. http://www.kfu.edu.sa, http://www.cu.edu.eg/, http://www.tanta.edu.eg in one of the Web pages of KFUPM's Web server, then these portal addresses will be added as a community member of the PA of KFUPM. This means, the PMAs of any PA will send friend of mine messages [Fig. 2] to the PA to register the portal addresses of the external links in their Web pages as a relevant one.

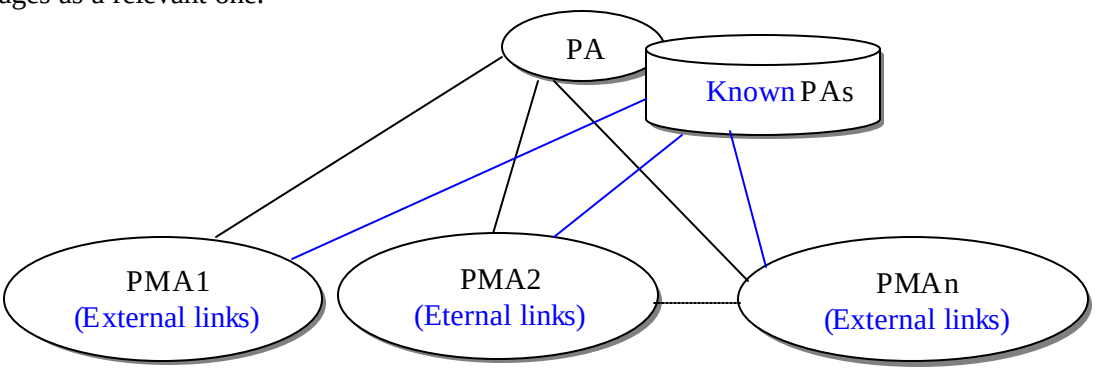

Figure 2 Portal agents community

Top

2.3 The Page Mining Agent

The PMA analyzes the data that are available on its Web page and continually keeps track of any changes in the content of its Web page. Each PMA starts with the base address when the PMA has got it from the PA. The PMA has its own parser, to which the PMA passes a URL, and an *SP* vector in which the PMA keeps all the policy keywords found in its URL. The PMA takes essential properties and principles given by the PA to create the *SP* of the PMA as an ontology that represents the context of the Web page as follows. The PMA sets the URL of its Web page as its name, loads, parses the HTML content of its Web page and extracts links, images, text, headers, applets, and the title. Then, the PMA eliminates the noisy words (non-informative words), stemming a plural noun to its single form and inflexed verb to its infinitive form. After that, the PMA creates its *SP* using an additional heuristics, in which additional weights are given to the keywords in the title and the headings of the Web page. Then, the created PMA registers itself to the PA and writes all the policy keywords into the *SP* file. A *SP* is used by the PMA to decide whether or not the keywords in the query belong to the PMA. The *SP* is a vector of important keywords, which are extracted and weighted by analyzing the contents of the Web page. Since the keywords are not all equally important for content representation of the *SP* vector of each PMA, an importance factor l is assigned to each keyword and decided by the kind of HTML tags, in which the keyword is included in the Web page. This means that the PMA will emphasize/de-emphasize some keywords based on the value of l. The PMA calculates the weight of the keyword and constructs its *SP* vector from the number of appearance (*tf*) and the HTML tags, which include the keyword within the Web page (e.g., in title, header, bold, italic), by using the equation $w_{ik} = l_k \cdot tf_{ik}$, where w_{ik} stands for the weight of keyword$_i$ in *k-th* HTML tag, and tf_{ik} stands for the number of occurrences that keyword$_i$ appears in *k-th* HTML tag. l_k stands for the weight decided by the kind of HTML tag$_k$ that includes the keyword$_i$ in the Web page. The total weight of a keyword$_i$ in the *SP* is the sum of all its weights in the HTML document of the Web page and is calculated by using the equation $w_i = \sum_{k=1}^{n} w_{ik}$, where n is the number of HTML tags within the Web page. The PMAs calculate the similarity between the Q_{in} and their Web pages based on the keywords they have in both of the *SPs* and the URLs of the PMAs. This similarity function is based on both Q_{in}-*SP* and Q_{in}-URL similarities. It is a hybrid similarity function that includes two components. At the retrieval phase, the PMA uses the *SP* to decide whether or not the user's query belongs to the PMA. The PMAs, when receiving a user's query from the PA, initiate the search process by interpreting the user's query and/or either asking, "*Is this yours*?" or announcing "*This is yours*" to its down-chain PMAs. The selected PMAs and/or their down-chain PMAs of each Web server interpret the user's query according to their *SP*s and reply the answer "*This is mine*" with a confidence value or "*Not mine*" with zero confidence. This confidence value depends on the similarity, where the similarity is based on co-occurrence of the user's query's keywords appearing in the *SP*s of the PMAs.

2.3.1 Clustering the PMAs into Communities

With the increase of the number of Web pages in the Web servers, it becomes better to cluster the Web pages into communities in order to find quickly the desired information. The PA clusters the PMAs into communities based on the similarity between the *SP*s and the incoming keywords of the Q_{in} over time in order to effectively focus on related subsets of PMAs. The Web server's administrator may define keywords as seeds for creating the base clusters of Web pages. The name of a cluster is initially constructed from the Q_{in} and the most common

Top

keywords in the *SP*s of the PMAs in the cluster and is dealt with the main attributes of the cluster. The cluster's name is updated according to newly input queries related to the cluster and a set of keywords surrounded by specific HTML tags included in the Web pages of the cluster and relevant to the queries. This means that, over time the communities of PMAs will be refined so that an agent may be assigned to or released from specific community.
We present the definition to create a cluster of PMAs as followings:

- Let Q be a set of cluster names $\{CNq_i \mid 1 \le i \le n, CNq_i = \{w_j \mid 1 \le j \le m\}\}$, where w_j a keyword, and n is the number of elements in Q. We call the number of elements in a set, size. Thus, n is a size of Q, and m is a size of CNq_i.
- Let Q_{in} be a user's query, where $Q_{in} = \{w_j \mid 1 \le j \le l\}$, l is a size of Q_{in}.

The algorithm of clustering the PMAs into communities is as followings:

1. Calculate the similarity between the keywords of the *SP*s.
2. Create a base cluster, each of which includes a keyword and its relative keywords, each of whose similarity with the keyword is 1.
3. Combine base clusters whose similarity value is over a threshold value.
4. When the user enters a query Q_{in}, the PA checks:
5. If $Q_{in} \cap CNq_j = \Phi$ for any $CNq_j \in Q$, then creates a new cluster Cq_i that consists of a set of Web pages relevant to Q_{in}, and Q_{in} is assigned to CNq_i, which is the name of Cq_i, i.e. $CNq_i \leftarrow Q_{in}$ and $Q \leftarrow Q \cup CNq_i$.
6. If $Q_{in} \cap CNq_j \neq \Phi$ for each $CNq_j \in Q$, and $Q_{in} \not\subset CNq_j$, then $Q \leftarrow Q \cup \{CNq_i\}$, $CNq_i \leftarrow CNq_i \cup Q_{in}$ and $CNq_i \leftarrow CNq_i \cup \{k_j\}$ for every $k_j \in Tag$. Where Tag is a set of keywords from the content of specific HTML tags in such Web pages that are in CNq_j and relevant to Q_{in}.

2.4 Router Agent and Routing Mechanism

Due to the size and growth-rate of the Web, a good distributed indexing/searching mechanism must be integrated with a distributed data-gathering mechanism. Although a single router agent is scalable enough to potentially handle thousands of PAs, in practice, it is desirable to run a separate RA for relevant PAs of a common topic, for instances, the PAs of AAAI, IEEE and ACM portals belong to one RA. The RA delegates the user's query to the most popular and relevant PA to the query to retrieve the Web pages, which are consistent with the user's information need. There is a router agent that holds a set of attributes that reflect the context of each PA [Table 1].

Registered PAs	Attributes of PAs			
PA_1	A_{11},W_{11}	A_{12},W_{12}	…	A_{1m},W_{1m}
PA_2	A_{21},W_{21}	A_{22},W_{22}	…	A_{2m},W_{2m}
…	…	…	…	…
PA_n	A_{n1},W_{n1}	A_{n2},W_{n2}	…	A_{nm},W_{nm}

Table 1 Portal agent's attributes

The PA_i sends its attributes $A_{ij}W_{ij}$, which are automatically determined from the cluster names of the PMAs communities in the Web portal, to the RA while the registration. Where A_{ij}

Top

means the *j-th* attribute of PA_i, and W_{ij} is its weight value, which assigned to each attribute and continually adapted by the RA over time based on the feedback from the IAs to reflect any changes in the context of the PA_i. Relevancy is used to determine the popularity of the PA for a particular type of queries. The RA maintains the relevancy S_j between Q_{in} and the attributes of PA_j using the equation $S_j = \sum_i w_{j,i} \cdot g(k_i)$, where $g(k_i)=1$ if k_i exists in both of Q_{in} and the attributes of the PA_j, otherwise $g(k_i) = 0$. There should be a single entity that controls the list of RAs. While registering a RA, it goes through one of several dozens of routers who work with, in turn, keeps a central database known as the router database that contains information about the profile of each router [Fig. 3]. Each of the routers has hundred of PAs and handles their requests. The compromise employed by distributed search consists of a set of routers, each of which handles the queries for a set of relevant PAs specialized in some way. The RAs register themselves to other routers to receive queries, which confirm to a particular context. The context defines the queries that a particular router will expect to be sent in. When the router receives a query from the IA, it does the followings:

- Looks for a list of relevant PAs, once the RA found relevant PAs.
- Assigns the query to specific PA because it already knew that this PA is relevant to this query, then merges the retrieved results and sends them back to the IA.
- Forwards queries to other routers if the results do not satisfy the user or the router could not find a relevant PA among its community of PAs. It may have to do this multiple times.
- Says, there are no relevant PAs to this query, or here is the most relevant router that may know more PAs about your query.

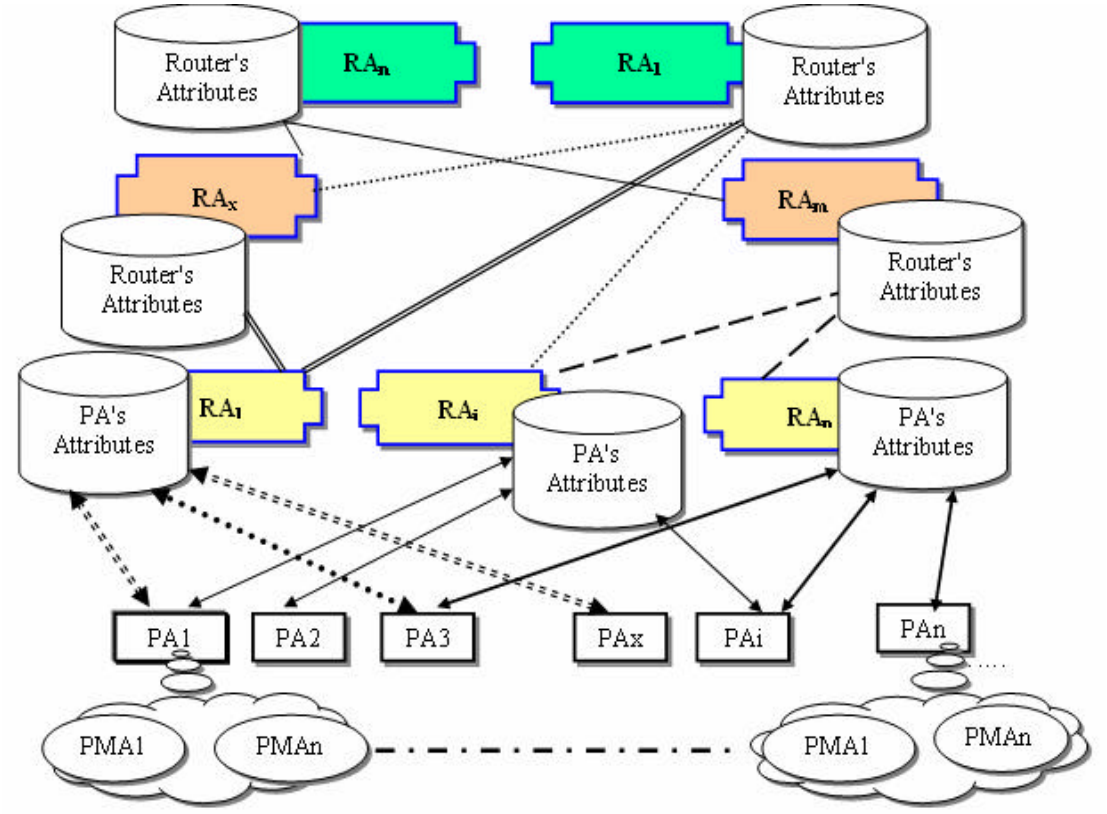

Figure 3 Routing mechanism

Top

3. System's Agents Communication

The system comprises the hierarchical structured agent communities based on a PA model. A PA is the representative of a community and allows all agents in the community to be treated as one normal agent outside the community. A PA has its role limited in a community, and another high-level PA may manage the PA itself. A PA manages all PMAs in a community and can multicast a message to them. Any PMA in a community can ask the PA to do multicasting its message. All agents form a logical world, which is completely separated from the physical world consisting of agent host machines. That means agents are not network-aware, but are organized and located by their places in the logical world. This model is realized with the agent mediator called Middle-Man Agent (MMA). MMA is primarily designed to act as a bridge between distributed physical networks, creating an agent communication infrastructure on which agents can be organized in a hierarchical fashion more easily. The system's agent consists of a communication unit and an application unit. The communication unit comprises the common basic modules shared by all agents, such as the community contactor, communication layer and message interpreter. The communication layer transmits data from source agents/machines to destination agents/machines through networks. In the system, we simply use TCP/IP, since most Internet applications implement their protocols on top of TCP/IP. The application unit comprises a set of plug-in modules, each of which is used for describing and realizing a specialized or connatural function of agents.

4 System's Agents Adaptation

There are several approaches that can be used to learn and adapt a UP by the IA [5, 6, 7, 28]. The IA creates a new list of keywords K_f for the feedback, $K_f = Q_{in} \cap K_s$. Where, $K_S = \{t_j \mid 1 \le j \le m\}$ is a list of keywords picked from the title and the headers of the selected URL, and $Q_{in} = \{q_j \mid 1 \le j \le l\}$ is the given query. If one of the keywords of the K_f does not exist in the query or the title fields of the URLs, which exist in the UP files, the IA adds the new keyword with an initial weight reflects the current user's response. By this way, the content of the UP files will evolve over time to reflect the actual UP.

The PMA allows only a relatively small change of the weights of *SP* based on the user's response, because adding/removing some keywords into/from the SPs may change the context of the Web pages. When the user says the page is interesting, the PMA changes the weights of the keywords in its *SP*, if and only if these keywords appear in Q_{in}, in order to make better match with the query to be entered next time. This means that, the PMA will emphasize/de-emphasize some keywords, frequently, reflecting the user's responses. Let *n* be a set of *SP* vectors and the user satisfied by the result retrieved by the PMA_i. Let Q_{in} be a user's query, such that $Q_{in} = \{q_j \mid 1 \le j \le l\}$, (l is a size of Q_{in}). The PMA_i changes w_{ij} , the weight of the keyword k_{ij} in the *SP* file, by adding a reward value $\Re$, so that the new weight, $w'_{ij} = w_{ij} + \Re * d_j$, where $d_j = 1$ if $k_{ij} \in Q_{in} \cap SP_i$ and $SP_i \in n$, other wise $d_j = 0$.

A weight value is associated with each attribute of the PA. The weights of the attributes that match the user's query are retrieved from the PA's attributes and updated (increased for positive feedback or decreased for negative feedback) by the RA based on the user's response to his/her query routing. As the contents of the Web servers are dynamic with new information being added, deleted, changed, and moved over time. Therefore, the attributes of the PA must also be dynamic with some attributes can be added or some existed

Top

attributes can be deleted to reflect well the context of the Web server. If the user gets satisfied with the retrieved answer from a specific PA while one of the query's keywords does not exist in the attributes of that PA, then the RA inserts that keyword as a new attribute with an initial weight that reflects the current user's response or adapt the existence one by using next equation, $w(t+1) = r \cdot w(t) + (1-r) \cdot \Re$ where r is a heuristic value and $0 < r \leq 1$.

5 Experimental Results

We have performed several experiments to make a consistent evaluation of system effectiveness. We mean here by the effectiveness, as it is purely a measure of the ubiquitously of the system, the ability of the system to satisfy the users in terms of the relevance of documents retrieved for user's queries, and the autonomously of the system's agents to adapt and learn over time.

Experiment 1: *in this experiment,* we measured how the RA is being able to adapt the PA's attributes over time and to get good correlation between the user's queries and the context of the Web portals. In order to understand the experiment, we define a Fitness value to show the correlation between the weights of the attributes calculated automatically by the RA (T) and the weights of the attributes calculated the system's administrator (S), as follows:

- System's administrator actual interest: $S_j = \sum_{k=1}^{m} b_k \cdot W_k$, where W_k is the weight of attribute k, and $b_k = 1$ if the administrator judges the keyword k in the PA_j as relevant for the context of the URLs of the Web portal, else $b_k = 0$.
- Adaptation of the attributes by the RA: $T_j = \sum_{k=1}^{m} W_k$.

We define the Fitness value $F_j = S_j / T_j$, which reflects the correlation between the two adaptations for PA_j. *In the experiment,* the user gave fifteen different queries. After frequent interactions of retrieval, the administrator checked the correlation of each attribute with the context of the Web portals and calculated S and T values, and then a Fitness value was calculated for the attributes of each PA. The Fitness values calculated after 10 and 20 retrieval interactions are shown in [Fig.4]. Figure 5 shows that the values of S and T are converging over time to each other. This means that the RA is being able to predict and adapt the PA's attributes to reflect well the context of the Web portal's over time.

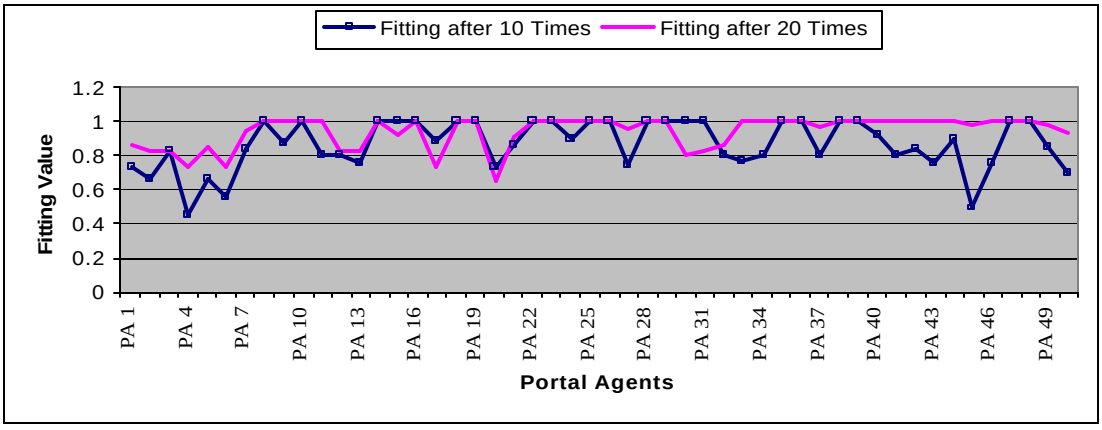

Figure 4 Portal agent's attributes adaptability

Top

Experiment 2: *In this experiment*, we attempted to evaluate the performance of the routing mechanism to retrieve relevant information to the user's queries from the PAs. We created fifty PAs that wrap fifty different Web portals; the PAs created the attributes and sent them to the router agent to be used in the routing process. The number of Web pages within the Web portals varies from 300 to 2500 pages. The users submit 30 queries to the IAs, which in turn forward the queries to the RA. The mean number of keywords per query, including the noise words, is 6.2. The RA delegates the user's queries to the most relevant PAs, receives back the results and forwards them to the IAs. Then, we calculated the precision of the retrieved URLs to the users' queries as the number of relevant pages retrieved divided by the total number of retrieved pages. The precision results depicted in [Fig. 5] show that the precision values vary from 0.7 to 1.0 and this means that the routing mechanism promises to achieve more relevant information to the user's queries.

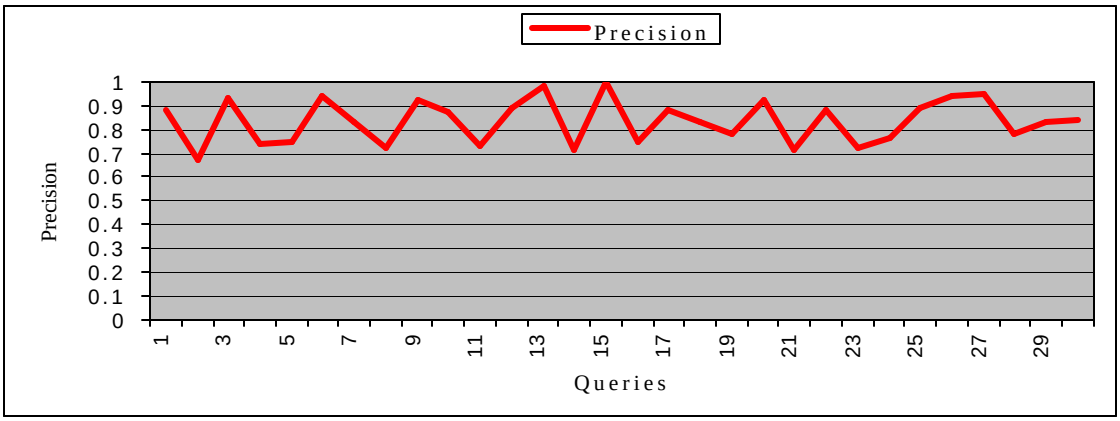

Figure 5 Precision of the queries submitted to the system

***Experiment 3*:** The topic domains of the Web pages are various and sometimes one Web page contains several topics at the same time. Therefore, the relationship among clusters is not simple and some clusters could have close similarity. We have implemented the clustering algorithm and measured how well the PA can cluster the PMAs of the Web portal into communities.

In this experiment, the data set for the experiment consists of about 2800 Web pages. The subject field of the experiment focused to the computer science-related Web pages. We have created PA that wraps of the IEEE (computer science society) Web portal. The PA creates the PMAs and the SP-file of this portal. The similarities between the SP vectors of PMAs are used to create the base clusters, and then the PA uses the algorithm described in section 2.4. After frequent interactions with the PA of the IEEE portal where the users submitted 65 different queries, the clusters were assigned by the similarity and the threshold for the assignment was 0.5. All the clusters were assigned if the similarity was greater than or equal the threshold value.

The results depicted in [Fig. 6] show that the system could cluster the PMAs of the IEEE Web portal into communities. The name of each cluster consists of the most common keywords in the *SP*s of its PMAs and the frequent keywords of the queries, which have been well answered by the PMAs of the cluster. Examples of some Web pages, which have been assigned into two clusters named "IEEE Computer society educational activities" and "IEEE conferences and publication", are presented. From the result, we could see that the pages in the same cluster do share similar topic and contents under the general query topics.

Top

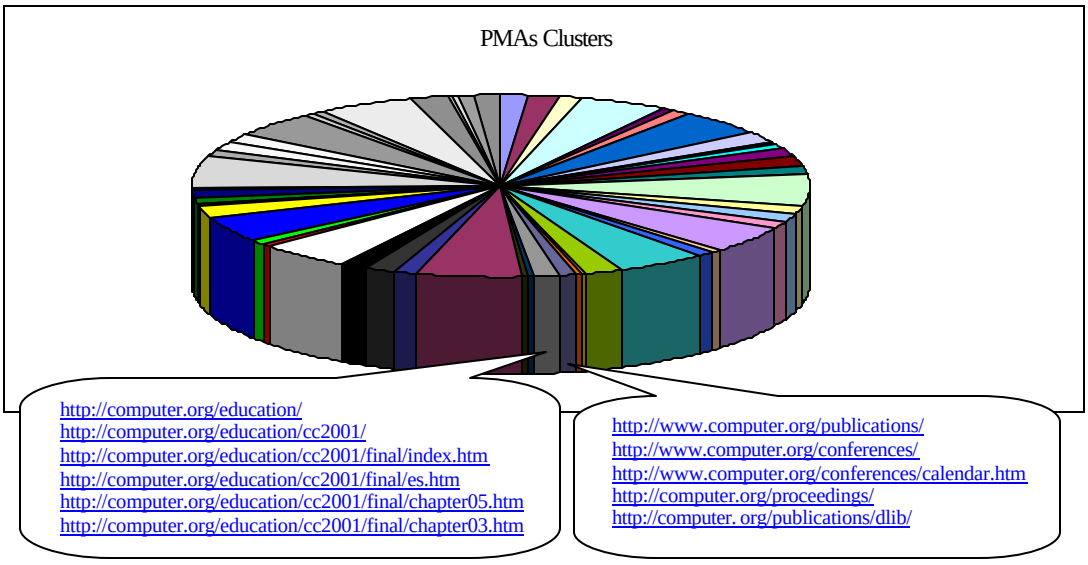

Figure 6 Clustering of the PMAs

6. Conclusions and Future Work

This paper discussed a multi-agent-based approach to build a ubiquitous information retrieval system. In this sense, the multi-agent system is composed of possibly large number of collaborative agents, which collectively try to satisfy the user's information need. The paper introduced methods to embed portal agents into the Web portals, to cluster the portal agents into communities and to route the user's queries to the most relevant PA. We created multiple PAs and developed a smart query routing mechanism for routing the user's query. We carried out several experiments to investigate the performance of system. Through these experiments, we ensure that system's agents learn, adapt and clustered into relevant communities over time. Currently, the routing of the system is designed as a simple query routing that binds to two hierarchical levels of RAs. There is a plan to scale up the system by increasing the number of PAs and develop more sophisticated routing mechanism for maintaining multiple hierarchies of RAs.

7 Acknowledgments

I would like to thank King Fahd University of Petroleum and Minerals for funding this research work and providing of the computing facilities, special thanks to anonymous reviewers for their valuable comments on this paper.

8 References

[1] Brusilovsky, P., Tasso, C., Special issue on user modeling for web information retrieval, "User Modeling and User Adapted Interaction", 14 (2004).

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

Top

[3] Bauer T., David B., "Real time user context modeling for information retrieval agents", Proceedings of the tenth international conference on Information and knowledge management, USA, pp. 568-570, 2001.

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

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

[6] Bonett Monica, Personalization of Web services: Opportunities and Challenges, Ariadna Issue 28, June 2001, ISSN: 1361-3200, http://www.aridane.ac.uk.

[7] Budzik J. and Hammond K., Watson: Anticipating and Contextualizing Information Needs, Proceedings of Sixty-second annual meeting of the American Society for Information Science, 1999, http://citeseer.nj.nec.com/budzik99watson.html.

[8] Chen Liren and Katia Sycara, WebMate: A Personal Agent for Browsing and Searching, Proceedings of the Second International Conference of Autonomous Agents, Minneapolis/ST, MN USA, May 9-13, 1998, pp.132-138.

[9] C. Ding, X. He, H. Zha, and H. Simon, "Web document clustering using hyperlink structures", Tech. Rep. CSE-01-006, Department of Computer Science and Engineering, Pennsylvania State University, 2001

[10] Dunja Mladenic, Text-learning and Related Intelligent Agents, IEEE Expert special issue on Application of Intelligent IR, July-August 1999, pp.44-45.

[11] Edmund S. Yu, Ping C. Koo, and Elizabth D. Liddy, Evolving Intelligent Text-based Agents, Proceedings of the 4th International Conference of Autonomous Agents, June 3-7-2000, Barcelona, Spain, pp.388-395.

[12] Helmy T., S. Amamiya, T. Mine and M. Amamiya, Agents Coordination-Based Web Infrastructure for Personalized Web Searching, Proceedings of the 2001 International Conference on Artificial Intelligence (ICAI-2001), pp. 97-103, June 25-28, 2001, Las Vegas, U.S.A.

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

[14] Helmy T., S. Amamiya and M. Amamiya, Pinpoint Web Searching and User modeling on the Collaborative Agents, LNCS Volume 2115, Issue, as a Proceedings of the 2nd International Conference on Electronic Commerce and Web Technologies EC-WEB 2001, pp. 305-314, September 4-6, 2001, Germany.

[15] Helmy T., S. Amamiya, and M. Amamiya, User's Ontology-Based Autonomous Interface Agents, The Second International Conference on Intelligent Agent Technology (IAT2001), A book entitled "Intelligent Agent Technology: Research and Development", World Scientific, pp. 264-273, 2001, Japan.

[16] Http://www.jxta.org/

[17] Hanna, K., Levine, B. and Manmatha, R., "Mobile Distributed Information Retrieval For Highly-Partitioned Networks," to appear in the Proceedings of IEEE ICNP 2003.

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

Top

[19] Kinuta, Y., Levine, B. and Manmatha, R., "Server Selection Techniques for Distributed Information Retrieval" CIIR Technical Report, 2003.

[20] Liu, X., and Croft, W. B., "Cluster-Based Retrieval Using Language Model", Proceedings of SIGIR '2004, pp. 186-193.

[21] Marian N., William B. and Anne H., Semantic Brokering over dynamic heterogeneous data sources in infoSleuth, Proc. of the internationa l on data Engineering (ICDE), 1999.

[22] Menczer F., Richard K., "Adaptive Retrieval Agents: Internalizing Local Context and Scaling up to the Web", Machine Learning Journal, Volume 39 , Issue 2-3 May-June 2000, Special issue on information retrievalPages: 203 – 242.

[23] Mine T., S. Lu, M. Amamiya, Discovering Relationship between Topics of Conferences by Filtering, Extracting and Clustering, Proceedings of the 3rd International Workshop on Natural Language and Information Systems (NLIS02), pp. 205-209, 2002.

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

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

[26] Somlo Gabriel. and Adele E. Howe, "Filtering for Personal Web Information Agents", *SIGIR'04*, July 25–29, 2004, UK, ACM 1-58113-881-4/04/0007.

[27] M. N. Huhns, "Agents as Web services" IEEE Internet computing v.6 n.4 p. 93-95, 2002.

[28] T. Louise, Su, "A comprehensive and systematic model of user evaluation of web search engines", Theory and background, Journal of the American Society for Information Science and Technology, v.54 n.13, pp.1175-1192, November 2003.

[29] Tarek Helmy, Satoshi Amamiya, Tsunenori Mine, Makoto Amamiya, "A New Approach of the Collaborative User Interface Agents", Proceedings of IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'03) , pp. 147-153, October 13-17, 2003 (Halifax, Canada).

Top