Intelligent Focused Agent for Building Databases from Distributed Web Systems

Rushdi A. Hamamreh **A.Mohsen K. Qawasmih**

Abstract—**This paper proposes a coordination model, for intelligent agents, based on filters and latent Semantic Indexing Algorithm to extract information from Web documents . This paper focuses on the development of an agent-oriented software engineering UML methodology. UML divide architecture of agent into roles to fetch, analyze and evaluate a document. If the filter recommends a document it, means that the document is relevant to the interest of owner. Also the filter coordinates agent work to fetch links from distributed web sites similar to interest of owner.**

Index Terms—**Agent, Information Retrieval, Latent Semantic Indexing, Adaptation, Unified Modeling Language (UML).**

I. INTRODUCTION

A continuous growth of internet usage, with billions of published documents and data distributed around the world, demands a useful, swift, precise and intelligent search system that satisfies users needs and queries. We noticed from IWS (Internet World Stats) website the rapid growth specially in the last few years. We notice that in Mar-2007 that the internet population grow up to 6,574,666,417 while the internet usage of latest data was about 1,114,274,426. This indicates the huge amount of documents that is distributed around the world [8],[1].

It's well known that search engines with centralized architecture can't index the whole Internet because of the exponential growth of published documents on the Internet.

A search engine with distributed architecture is a scalable solution to this problem [3].

The study of multi-agent systems in the field of Distributed Artificial Intelligence (DAI) began about 20 years ago. Today these systems are not simply a research topic, but are becoming important in academic, industrial and commercial applications [5],[7].

Our agent is expected to establish new cooperation among research groups in relevant areas but also to strengthen existing contacts and efforts for research and development of intelligent information agents.

Rushdi A. Hamamreh is with Department of Computer Engineering, Al-Quds University. (email: rushdi@eng.alquds.edu)

A.Mohsen K. Qawasmih is with Department of Computer Engineering, Al-Quds University. (email: aqawasmi@eng.alquds.edu)

Our system which is based on Intelligent Information Agents aims at helping the user, melting together the multi-agent system (MAS) and information access technologies by investigating the extent methods of Artificial Intelligence, Database Systems and Information Retrieval (IR) can be applied to information discovery by themes on the Internet and the World Wide Web.

Within the framework of our suggested architecture, we use a set of topic target databases (collections) of electronic documents published in the Internet. These databases belong to different owners who are responsible for content, indexing and quality of search. Administrator's demand is automatically propagated to one or more databases with topics relevant to the target topic [2],[4].

II. AGENT VIEW

An agent is a computational entity such as a software program or a robot that can be viewed as perceiving and acting upon its environment; it is autonomous in its behavior since it partially depends on its own experience. As an intelligent entity, an agent operates flexibly and rationally in a variety of environmental circumstances given its perceptual and effectual equipment. Behavioral flexibility and rationality are achieved by an agent on the basis of key processes such as problem solving, planning, decision making, and learning [11],[2]

Managing and controlling such networks, the services they provide, and the communications they involve are crucial to keep Internet a useful future tool. However, there is a growing awareness that current centralized IR architectures will soon reach the limits of their scalability. We argue that distributed but coordinated mechanisms that support adaptation and self-optimization of Information Agent Societies can be an answer to this problem[8],[4].

III. AGENT ARCHITECTURE

In a distributed agent framework, we conceptualize a dynamic community of agents, where multiple agents contribute services to the community. When external services or information are required by a given agent,

instead of calling a known subroutine or asking a specific agent to perform a task, the agent submits a high-level expression describing needs and attributes of the request to a specialized Facilitator agent. The latter will make decisions about which available agents are capable of handling sub-parts of such request, and will manage all agent interactions to handle the complex query.

The advantage of such distributed agent architecture allows the construction of systems that are more flexible and adaptable. Individual agents can be dynamically added to the community to extend the functionality that the agent community can provide as a whole. The agent - system is also capable of adapting to available resources in distributed environment.

Using AUML, we will capture the MAS complexity by role decomposition and control MAS environment dynamicity by role/agent entities separation. In terms of modeling, AUML supports the idea of UML extension toward Agent UML, which results in the integration of agent classes, role classes and interaction protocols to UML [10],[2].

Figure 2 represents the architecture of an agent with next roles:

A. Role A (Document Fetcher)

This Agent Role uses "**WGET"** utility for document downloading. The link of this document is taken from a storage volume which contains a queue of links to be fetched. Links queue starts from a set of start Links presented by the administrator. Every Link is assigned an estimation of usefulness for seeking new relevant documents. At first, the newly included to this queue Link is assigned (ranked) number 1 according to its usefulness.

The next stage of this role is Stemming. It is logical view of documents from full text to a set of indexed terms. This stage includes Accent spacing, Noun grouping, Stop words removing until it reaches index terms of a full text.

Then, the index terms of the fetched document will be handed to Agent role C, which is responsible for figuring out whether the document is relevant or not.

If the document is relevant, Agent role A starts to extract all links from this document because the probability of relevance of these links is high. These links are handed directly to Agent Role D.

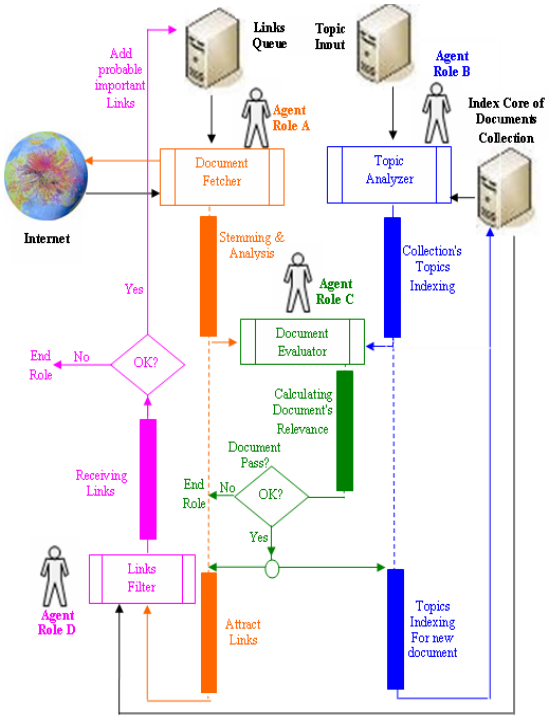

Figure 2. Agent architecture AUML

B. Role B (Topic Analyzer)

Using PLSI arithmetic method, this Role is responsible for two major stages:

Stage 1, includes receiving main target topic terms that are produced by the Administrator and stored in the topic input volume. PLSI method is used to give a weight (value) to these terms, in addition to index topic terms that comes from relevant documents stored in the Index core Documents collection. Target topics terms weights are continuously modified when a new relevant document is added; at the same time these modifications are saved in Topic Input storage volume, and this loop increases the smartness of the agent. The modifications are also handed to Agent Role C, which is responsible for figuring out if the document is relevant or not.

Stage 2, starts when Agent Role C decides that the fetched document is relevant, it starts to analyze the topics of document index terms using PLSI before adding the relevant document to the collection.

C. Role C (Document Evaluator)

This role is responsible for receiving index terms from Role A, and Topic index from Role B, and starts to calculate the relevance of the document, and its weight. Then it determines when holding a comparison with a target threshold, whether the document is relevant or not. If the result is positive, then both roles A and B start their mission on this new gift. However, for negative result, Role C ends its job on this useless document.

D. Role D (Links Filter)

At every next step Role A chooses from the queue a Link with maximum value of estimation of its usefulness, downloads and evaluates it. If this document is accepted by evaluator, then at next step, the agent randomly chooses links presented in its text and includes them into Links queue with usefulness estimation equal 1. If a downloaded document is not accepted by evaluator, then estimation of the usefulness of its Link where it occurs, is decreased. As a result, estimation of the Link's usefulness is an approximation of probability of relevance of a link from the document to the collection topic.

This role has to make sure that all attracted links are useful and not repeated (already checked before) in order to increase performance. This is done with help of the stored documents collection description. So every link has to be filtered and the role decides whether to add it to the Queue or not (which means to end role).

IV. INFORMATION RETRIEVAL MODELS

Using Probabilistic Latent Semantic Indexing (PLSI) arithmetic method, both Role B and C are responsible for analyzing the entire set of documents from this collection and create the collection description which reflects the main subjects presented in it. We have used for this a proposed probabilistic latent semantic indexing [3],[5].

The goal of the latent semantic indexing is extraction of latent factors which reflect a set of narrow topics presented in a given collection.

Let $z \in Z = \{z_1,\ldots, z_k\}$ be set of these factors, and `term frequency tf`. Let denotes:

- $P(z_i)$ – probability that randomly selected document from the collection that best corresponds to the topic z_i.
- $P(d|z)$ – probability that for the given factor z_i this factor best corresponds with the document d_i .
- $P(w|z)$ – probability that for the given factor z_i this factor best corresponds with the word w_j.

Here $d \in D=\{d_1, \ldots ,d_n\}$ is set of all documents from the collection and $w \in W =\{w_1,\ldots, w_m\}$ is set of all words from this collection.

Functions $P(z_i)$, $P(d|z)$ and $P(w|z)$ can be estimated in the process of a likelihood function maximization. This function is presented in the following form

$$L = \sum_d \sum_w tf(d,w) \log(P(d,w)),$$

Standard Expectation Maximization algorithm is used for maximization of this function. Two steps are executed at every iteration of this algorithm. The first one is Estimation

$$P(z \mid d,w) = \frac{P(z)P(d \mid z)P(w \mid z)}{\sum_{z'} P(z')P(d \mid z')P(w \mid z')}.$$

The second one is Maximization

$$P(w \mid z) = \frac{\sum_d tf(d,w)P(z \mid d,w)}{\sum_{d,w'} tf(d,w')P(z \mid d,w')},$$

$$P(d \mid z) = \frac{\sum_w tf(d,w)P(z \mid d,w)}{\sum_{d',w} tf(d',w)P(z \mid d',w)},$$

$$P(z) = \frac{\sum_{d,w} tf(d,w)P(z \mid d,w)}{\sum_{d,w} tf(d,w)}.$$

To generate the collection filter we have selected the most heavy words from W. Weight of the word w is calculated as:

$$\text{weight(w)} = \sum_{z \in Z} P(z)P(w \mid z)$$

The key idea of LSI [1] is to map documents to a vector space of reduced dimensionality, the latent semantic space.

This mapping is computed by decomposing the word(w_m) / document (d_n) matrix $D = \tilde{N}$, with singular value decomposition (SVD), $\tilde{N} = U \Sigma V^T$, where U and V are orthogonal matrices $U^TU = V^TV = I$ and the diagonal matrix Σ contains the singular values of N.

The LSI approximation of N is computed by thresholding all but the largest k singular values in Σ to zero ($=\Sigma$), which is rank k optimal in the sense of the L_2-matrix norm as is well-known from linear algebra, i.e., one obtains the approximation $\tilde{N} = U\Sigma V^T \approx U\Sigma V^T = N$. The same representation applies to queries q, $q=q^T U_k (\Sigma^\kappa)^{-1}$. Note that the L_2-norm approximation does not prohibit entries of $\tilde{N}$ to be negative.

Let us rewrite the aspect model matrix notation. Hence define matrices by $U=P(d_i|z_k))$, $V=P(w_j|z_k)$ and $\Sigma=\text{diag}(P(z_k))$.

There are two different approaches to extract topics from document, as we see from the outcome of experiments, the focused agent had different behaviors

V. AGENT ADAPTATION

The goal of using PLSI method is to analyze the whole set of administrator's queries which reflects information need. This analysis can be used to find new subjects which are interesting to the administrator but are poorly presented in the collection core.

In order to do so, we have used the following approach to build . At first graph G of all words used in the administrator's terms was created.

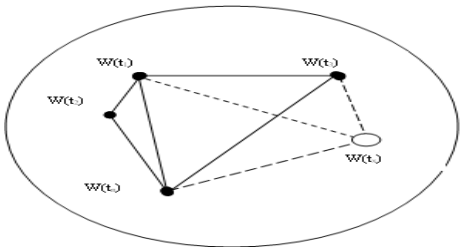

Figure 3: graph G , weight of term $w(t_{1..n})$, weight of new term $w(t_x)$.

Every word is presented as a vertex of this graph. Two vertices are joined with an edge if and only if the pair of corresponding words occurs in the same query. Every vertex should have a weight w(t) which reflects the role of this word in the collection subject. Some of these words are presented in the collection core and we can use probabilistic latent semantic indexing to calculate their weights. But a part of words, presented in the queries, can be new (not presented in the collection core). To estimate the weights, we've used the following method.

We have supposed that the weight of every new word should be equal to the average value of weights of words which are neighbors to this word. After new word, this algorithm will estimate the weights of all new words according to this proposal and adapt it to its owner.

$$W(t_x) = \frac{\sum_{g=1}^{n} W(t_g)}{n}$$, n : number of all vertex.

All information about queries words and their weights is stored as queries statistics**.**

VI. APPLICATION

For our architecture, we implemented simple interface (see figure 8) , which determines the six options: directory of core collections, path of file queries, directory of new files threshold of two filters and number of topics.

VII. EXPERIMENTS AND CONCLUSIONS

In this section, we will present our experiments with focused agent. We will measure Relevance (precision) and coverage (recall).

The evaluator (Role C) needs two values to make a decision that the document is relevant or not, the first value is the number of words m in each topic and the second value is the number of topics k in each document from the topic Analyzer (Role B). After downloaded 5000 documents from internet , with 50 start links and 200 relevant documents in the core collection which were selected by administrator.

The evaluator also had given also two optimal values for 0.1 filter queries and 0.5 for filter collection [13].

The results are shown in Figure 4 and Figure 5. The two topics we chose are :Information Security (ISec) and Medical (Med).

This means that our agent downloaded a set of new documents; the agent has different behaviors depending on the method of extract data and type of topic, the method of extract of topics from documents can retrieve the relevant information if a selection of the optimal numbers of topics that contain in the each document, and the number of terms (words) that contain in the each of topic.

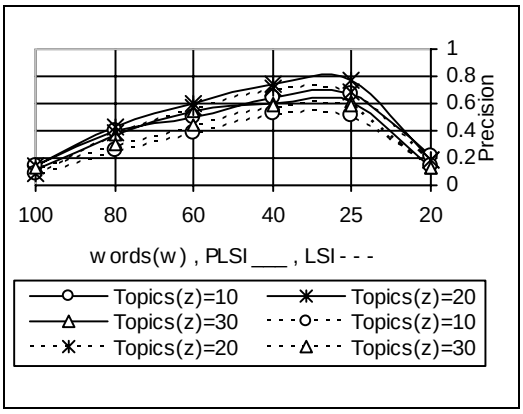

Figure 4. Precision($\mathbf{w_m}$, $\mathbf{z_k}$) curves for the two tests databases with LSI and PLSI.

The agent recommended about 79 % of downloaded documents with 20 topics and 25 words for PLSI indexing and 69% of downloaded documents with 20 topics and 35 words for LSI indexing; this mean that LSI needs more word than PLSI to define the optimal numbers of topics and words, see the Figure 4.

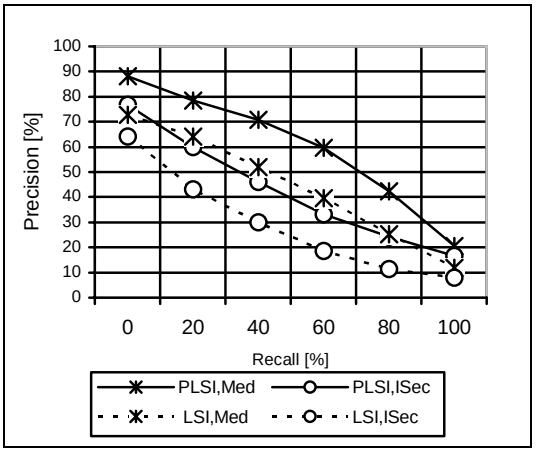

Figure 5. Precision(recall) curves for the two tests databases with LSI and PLSI.

The agent's coverage of topics that have words with minimum synonyms is better than the topics that has may synonyms, see the Figure 5.

The experiments have consistently validated the advantages of PLSI over LSI. Substantial performance gains have been achieved for 2 data sets and both term weighing schemes.

Probabilistic Latent Semantic Indexing which achieves significant gains in precision over LSI. Figure 6 and Figure 7 shows precision and recalls agent after adaptation is increased by about 4%.

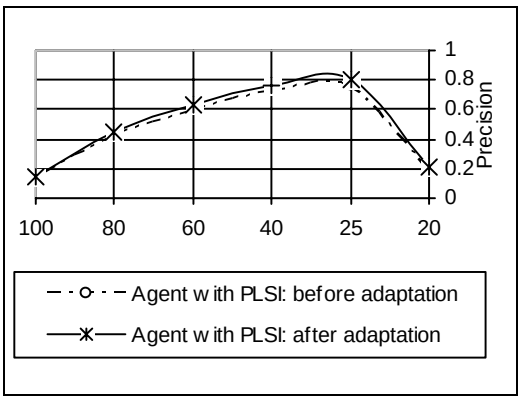

Figure 6. Precision(w_m , z_k) curves for the two test databases before and after adaptation.

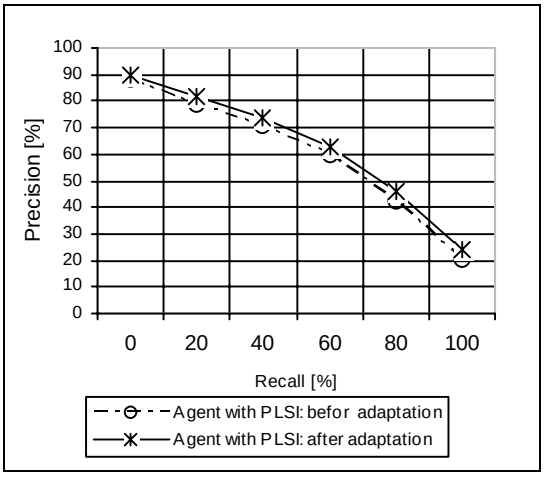

Figure 7. Precision(recall) curves for the two tests databases before and after adaptation.

VIII. APPLICATION

Our agent has simple interface; it is to setup needs to define directory source files of core collections, path of file quires, directory of new documents, threshold of two filters and number of topics.

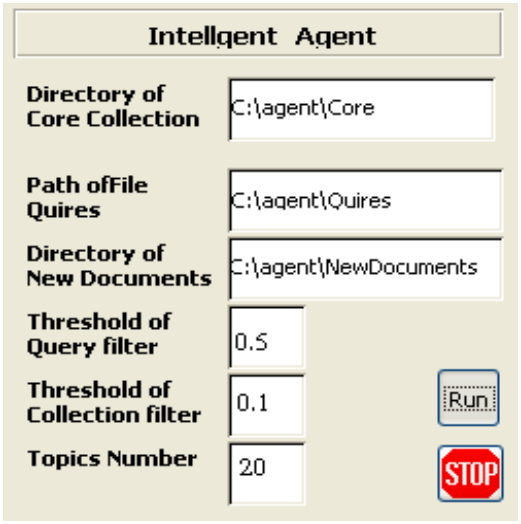

Figure 8. Application of Intelligent Agent

REFERENCES

[1] Lan Huang , Survey On Web Information Retrieval Technologies, Computer Science Department, State University of New York at Stony Brook, 2000.

[2] Junghoo Cho and Lawrence Page, Efficient crawling through url ordering, Proceedings of the 8th International WWW conference, Canada, Tronoto, May1999.

[3] Probabilistic Latent Semantic Analysis, Thomas Hofmann, Stockholm, UAI '99

[4] Rushdi Hamamreh. Multi-agent system of a distributed environment to build thematic collections // Web Instruments, № 9, 2001.

[5] S. Haverkamp, Intelligent information Agents, JASIS, V49, N4, 1998.

[6] Hofmann T., Latent class models for collaborative filtering. In Proceedings of the 16th International Joint conference on Artificial Intelligent, 1999.

[7] Michael Wooldridge, An introduction to Multiagent Systems, John Wiley & sons Ltd,2002.

[8] Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence, edited by Gerhard Weiss, The MIT Press Cambridge, Massachusetts London, England,1999.

[9] Towards Agent Societies for Information Retrieval, Holger Billhardt and Sascha Ossowski Universidad Rey Juan Carlos Department. of Computer Science, 2004.

[10] World internet usage and population statistics, IWS (Internet world stats) website www.internetworldstats.com.

[11] Agent Role Locking Theory ARL, Salaheddin J. Juneidi, George A. Vouros Department of Information and Communication Systems Engineering School of Sciences University of the Aegean Samos, Greece, 2004.

[12] Toward Programming paradigms for agent oriented software engineering Department of Information and Communication Systems Engineering School of Sciences University of the Aegean Samos, Greece Salaheddin J. June 2004.

[13] Autonomous agent for gathering information to build focused index from distributed environment, International Journal of Computer Science and Network Security, January 2008.