IJICT International Journal of Information & Communication Technology

ITRC

Volume 2- Number 4- September 2010 (69-78)

Research Note

Improving the Performance of Proposed Multi-Agent Domain Specific Search Engine Using Query Refinement Component

Elmira Hajimani
Electrical and Computer Eng. Faculty
Shahid Beheshti University
Tehran, Iran
Elmira.h.mani@gmail.com

Eslam Nazemi
Electrical and Computer Eng. Faculty
Shahid Beheshti University
Tehran, Iran
nazemi@sbu.ac.ir

Received: June 12, 2010 - Accepted: August 3, 2010

Abstract— **The web size is increasing continuously. The more the Internet is growing; the more tendencies the people have to use the search engines. Moreover, since most of the commercial search engines are based on keyword indexing, there are many records in their result lists that are irrelevant to the user's information needs. It is shown that for retrieving more relevant and precise results, the following two points should be concerned: First of all, the query (either it is generated by a human or an intelligent agent) should be expressed in an accurate and exact manner. Second, we should empower search engines with the ability to capture the semantic relation between the words and the query context. Hence, different search engine architectures, each of which containing query refinement or semantic understanding components, have been proposed. Each architectural model has its own specific properties; but, most of them focus on only one of the two points mentioned above to improve the overall system efficiency. Moreover, in existing architectures, query refinement components have direct interaction with users which may either take their time or threat their privacy while gathering basic information. In this paper, we proposed an improved architectural model for agent and ontology based search engine which uses domain ontology for semantic understanding and a query refinement subsystem based on fuzzy ontology. This subsystem helps Search Agents to refine their queries, express them in a more precise way and get more relevant results. The simulation result shows that using this query refinement subsystem by Search Agents can improve the system efficiency up to 5.2%.**

Keywords- Search engines, Domain ontology, Fuzzy ontology, intelligent agents, query refinement

I. INTRODUCTION

In our complex world, the relation between entities plays an important role in describing or understanding them. In fact, to comprehend what an entity or a concept is; we usually consider the ways it relates to other entities or concepts.

Whenever a query is given to a keyword based search engine, there are plenty of records in its result list that do not meet the user information needs. The reason is that there is no way to register the semantic relation between the words in the current web architecture so while entering keywords to these kind of search engines, the semantic relations between them are omitted.

This problem motivated researchers to help people by following two different strategies [2]:

- Changing the infrastructure of the current web to the semantic web.

- Placing the keyword based search engines as the base and doing some modifications to make them considering the query and web page context in order to improve their efficiency.

There was a big problem over the realization of the first idea. The problem was that there were already millions of millions documents in current web that should apply considerable modifications in their structure to express their content in RDF and RDFS [9]. On the other hand, for solving the problem of word sense ambiguity (one word corresponding to several different meanings and vice versa) and making a common understanding in a specified domain, diverse domain ontologies should be developed to cover existing documents in www. As a result, the growth of semantic web slowed down and most of the search engines that were developed on the basis of the semantic web remained in research laboratories. It is why that QuizRDF combines traditional keyword querying of WWW resources with the ability to browse and query against RDF annotations of those resources [1].

Search engines that are following the second strategy use keyword based search engines as their underlying layer and then add additional components to enhance their recall and precision [4,5,6,10]. Our proposed architecture in this paper also follows the second strategy. It assigns the defined tasks to intelligent agents to enhance the ability of parallelism as well as to improve reliability and distribution level. Moreover, using domain ontology helps agents to better understand the concepts of the human world and their relations and do their tasks more precisely.

The rest of paper is organized as follows: section II reviews the related works briefly. Section III will describe the proposed architecture in detail. Section IV elaborates how to construct the fuzzy ontology which is used in query refinement subsystem. Experimental results are found in section V. Section VI concludes the paper and presents future works.

II. Related Works

Different techniques have been applied to improve search engines' performance which involves finding out about who the user is; what his preferences are and what his search context could be through his activities. Furthermore using intelligent agents, applying ontologies, presenting combined architectures and creating domain specified search engines are the other ways that have been proposed to improve the performance and efficiency of traditional search engines as well as satisfying the users of the this ever growing web.

Information retrieval systems such as InWiss, QuizRDF and OWLIR have used ontologies to import human world's concepts and their relations in a search process [2]. These architectures do not take the advantages of neither using intelligent agents nor query refinement components and focus only on query vagueness problem and use semantic recognition techniques to solve it.

According to [3] which has introduced Agent Space architecture for search engines, one of the most important factor in enhancing search engines' performance, fault tolerant and scalability is the use of intelligent agents. This architecture mainly concentrates on parallelism and workload balancing through distributing tasks on several machines and uses intelligent agents for its goal. Agent Space architecture does not concern about query ambiguity and does not use neither query refinement nor semantic recognition components.

Master-Web and AGATHE are the two combined patterns introduced in [4], [5] and [6]. These two information retrieval systems apply keyword based search engines as their underlying layer and use ontology on one hand and intelligent agents on the other hand to not only consider the semantic of the query terms but also provide a good level of reliability, modularity and parallelism. These two architectures also have no query refinement mechanism to help neither the users nor the Search Agents which interact with the keyword based search engines.

The approach in [14] also introduces a personalized search engine which re-ranks the URLs appeared in the result list of an underlying search engine according to the information gathered from users' profiles. To do this task, the user profile including web pages that already visited, are collected. In next step, an Enriched Fuzzy Concept Network (EFCN) is automatically generated based on a predefined concepts vector and the user profile using WordNet ontology as a part. After that, it acquires the fuzzy relation between each appeared document in the result list and the concepts that exist in EFCN; WordNet ontology also is applied here to enrich the outcome. Finally, it uses a ranker algorithm to obtain the ranking of each document and reorders documents based on their rankings. The main problem of this proposed personalized search engine is that it uses a predefined concept vector as its input which is actually should be defined by each user based on his frequent information needs. That is a time consuming task which most users are reluctant to do.

According to [7] PASS search engine uses a fuzzy ontology to help users to refine their queries and getting more relevant results trough the keyword based search engine. The ontology is built automatically and determines the fuzzy relation between the terms. In PASS architecture the fuzzy ontology is constructed from a collection of documents which are not collected based on domain ontology so the flexibility of the system on changing the domain is reduced.

There is an ontology based query expansion method, introduced in [13], which is especially developed for vertical search engines. In this method, a domain knowledge database is first established by adopting ontology as the describing tool. The method then constructs the semantic diagraph with each query-term as the first vertex based on the domain knowledge. In next step, it calculates the semantic distance between the first vertex and each vertex in the semantic diagraph and then selects the expanded terms of each semantic diagraph by considering a threshold. Finally, this method uses a logic operator to combine

all the terms gotten from the semantic diagraph and obtains the result of query expansion. As indicated in [13], there are some parameters in semantic distance formula that there is no determined method to set them properly.

In [11] BaalaMithra also uses domain ontology to present a context-aware automatic query refinement method which performs a contextualization and conceptualization on the user clicked web-pages to extract information to disambiguate the query term and identify the context of the query terms. The conceptualization phase of this method also calculates the semantic distance between the queries, contextualization suggested terms and the domain ontology in order to suggest the semantically relevant concepts.

In [12], a semi supervised query expansion method with minimal feedback is presented. This method asks user to identify at least one relevant document among the list of retrieved results by a search engine. It then applies a machine learning technique based on the transduction called Spectral Graph Transductor (SGT) that creates classification labels (including relevant and irrelevant labels) for the rest of the retrieved links in the result list. The method then calculates a score for each term of the documents labeled as relevant ones. Finally, it selects the terms with the top m scores to expand the initial query. As indicated in [12], the computational time for this method increases linearly with the number of unlabeled documents which should be concerned while using this method in practice.

HQE, a hybrid new approach for query refinement, is introduced in [8]. This method applies ontology to find similar users and then uses the RBF neural networks to extract the most relevant documents and related terms through the queries of similar users.

Our proposed architecture for domain specified search engines tries to improve the performance of search engines by providing high level of distribution, modularity, reliability, recognizing the semantic relation between the words as well as refining the queries by adding extra terms to them. These extra terms help queries to point more specific to a subject. All the mentioned properties are gained by using intelligent agents, domain ontology and query refinement component in the proposed architecture respectively.

III. PROPOSED ARCHITECTURE

This section is dedicated to elaborate our proposed architecture for agent based domain specific search engines. Our main idea is about considering multiple aspects that have considerable impact on presenting precise results at the same time. By precise results we mean information which fits to the user information needs in a specified domain. To achieve our goal, we concentrated on the repository construction part of the search process and presented an architectural model which applies intelligent agents, domain ontology, fuzzy concept and a query refinement mechanism all together to not only make the repository content more precise and domain related; but also help the repository construction process more reliable, modular and distributable.

What makes this architectural model different from all other existing architectures is a query refinement component which helps Search Agents to refine their queries and express them in a more precise way while interacting with their underlying layer. This query refinement subsystem is applying fuzzy ontology to help the Search Agents. This subsystem has two major differences with existing query refinement components in other architectures:

- **Gathering basic information:** The query refinement component needs some source of information to propose new terms in order to refine a query. In our proposed architecture we use domain ontology to gather the basic information, in other words, on the basis of the concepts that are already defined in the domain ontology; the query refinement component looks for the web pages that are related to these concepts and stores them in a database for further processing; as a result, in spite of other existing query refinement components, there is no need to violate the user privacy through monitoring his behavior or his files to know about his preferences. Furthermore, users do not need to take their time to fill out the forms to introduce themselves and their preferences.
- **Selecting appropriate terms from created data base to refine queries:** In existing architectures, query refinement components have to interact with users for selecting appropriate terms from database and adding them into a query; whereas in our proposed architecture, the proposed fuzzy ontology constructor subsystem calculates the fuzzy relations between terms which are extracted from the stored web pages in previous step automatically and then suggests the terms with highest membership degree to refine the query of each Search Agent.

The following is an introduction to the architectural model's components as well as how they relate to each other and the workflow.

A. *Intelligent Agents*

Our proposed architecture for domain specific search engines, illustrated on Fig. 1, is a multi-agent system which assigns defined tasks to different intelligent agents to gain acceptable level of parallelism and distribution. These agents are: Supervisor Agents, Search Agents, Page Object Maker Agents, Storage Agents and Term Extractor & Qualifier Agents. Term Extractor & Qualifier Agents are only used in fuzzy ontology creator subsystem.

B. *Ontologies*

There are two ontologies in our proposed model. The first one specifies the concepts which exist in the search domain; it also includes the relation between the concepts. The domain ontology plays an important role in creating the facts of the knowledge base. These

facts are then used by the inference engine. The second ontology is a fuzzy ontology which helps Search Agents to add more specific terms to their queries and get more relevant results while interacting with keyword based search engines.

C. *Databases*

The facts are produced based on both the domain ontology and the web pages which are retrieved by the Page Object Maker Agents. These facts are then saved in a temporary database which is located in working memory and called "Facts composed from retrieved web pages DB". The classification rules are also defined based on the domain ontology and saved in a database named "Classification rules DB". This database is readable by the inference engine.

The fuzzy ontology which is used by the Search Agents is stored in a database called "Fuzzy Ontology for Query Refinement DB". The fuzzy ontology constructor subsystem creates this fuzzy ontology using the domain ontology. The "Classified Pages DB" is used to store classified web pages. These web pages are then accessible through the user interface.

D. *The inference engine*

The inference engine first classifies the web pages based on the existing facts and classifying rules and then calls the Storage Agent which is related to that class to store the classified web page in the database.

E. *The shared space*

The Page Object Maker Agents retrieve and process web pages and create an object for each of them. These objects are then saved in a shared space in working memory in order to produce facts from them. These facts describe the created objects and act as an input to inference engine.

F. *The user interface*

This interface enables the interaction between the users and the database which contains the classified web pages that are categorized based on the domain ontology. Users send their requests to the system through the user interface and get the related URLs.

G. *How the components relate to each other and the workflow*

The domain ontology consists of some classes and their relationships. Each class stands for a concept in the domain. For each class in the domain ontology, a series of related terms are defined. For each class of the ontology, the Supervisor Agent creates and initializes a Search Agent. The initialization includes informing the Search Agent about what it should look for.

After initializing, each Search Agent queries the fuzzy ontology database to refine its query. In this step the query terms will be complemented by more specific and narrower terms. The Search Agents then pass their refined queries to a keyword-based search engine and receive the most top ranked URL results which are in the form of HTML pages. For each URL result, the corresponding Search Agent creates a Retrieve & Page Object Maker Agent and initializes it with the URL.

The Retrieve & Page Object Maker Agent then connects to the web, retrieves and processes the content of the webpage in order to find important phrases which are related to the domain ontology. The process includes omitting the stop words, stemming, extracting the important and domain related phrases. The Agent finally reorders the selected phrases based on their occurrences in the webpage and selects the most N frequent related terms as the keywords of that webpage. Retrieve & Page Object Maker Agent then creates an object from that webpage (including both webpage content and obtained information) and places it in the Shared Space which is in working memory.

The objects are then translated to the facts and will be placed in a knowledge base. The classification rules are also written using the domain ontology. For example, the classification rule in Fig. 2 indicates that if the terms like "information" and "data" exist among the keyword list of a webpage object, the webpage will be classified as a Data Base related webpage.

The inference engine, first, classifies web pages using both facts and classification rules and then calls the Storage Agent of each class to store its corresponding classified webpage in a permanent database. For each class in the domain ontology there is a Storage Agent. These Storage Agents are initialized by the inference engine and have the duty of storing classified pages in the database; hence the classified pages will be accessible to users through the user interface. For example, as showed in Fig. 2, a Storage Agent named DB_Class_StorageAgent is activated to store the database related webpage as the rule fires.

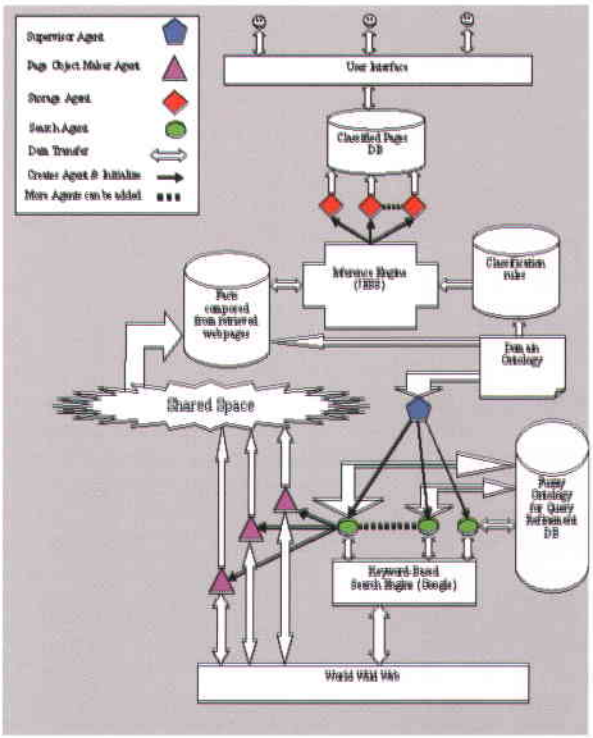

Figure 1. Our proposed architecture

IV. THE PROPOSED FUZZY ONTOLOGY CONSTRUCTOR SUBSYSTEM

The purpose of this subsystem is to arrange phrases in a hierarchical manner, hence whenever a Search Agent consults with this subsystem in order to refine its query and express it in a more precise way, the subsystem can determine which terms are participating in Narrower Than (NT) relationship with the terms that are appeared in the query and suggests the appropriate narrower terms to the Search Agent. For producing fuzzy ontology of term associations, a source of domain related documents is needed. Our proposed fuzzy ontology constructor subsystem, in detail represented in Fig. 3, is a multi agent subsystem (MAS) which automatically collects all needed documents based on domain ontology and then extracts important key terms for further processing. All formulas used for calculating the fuzzy relationship between the domain related terms are according to literature [7].

According to the defined concepts in domain ontology, Supervisor Agent creates a number of Search Agents; each of which relates to a specific concept. Search Agents are then initialized by the key terms of their corresponding class. They are in charge of querying the keyword based search engines in order to get the most top ranked URLs related to the concepts they are supposed to search for.

For each retrieved URL, Search Agent creates and initializes a Term Extractor & Qualifier Agent which performs its task in three consecutive steps: At first, it connects to the web to retrieve the content of the page and second, it utilizes the WordNet -a linguistic database formed by synsets- API to extract key terms and their frequency occurrences. Finally, validated key terms along with their frequency occurrences and the related URL (where these key terms are extracted from) are stored in a temporary table in database.

Suppose that $C=(u_1,u_2,...,u_n)$ is a collection of URLs u_i where each URL $u=(t_1,t_2,...,t_m)$ contains a set of key terms t_j. Given a term t_j, the occurrence of t_j in URL u is represented by $occur(t_j,u)$ and its membership value is defined by $\mu_{occur}(t_j,u)=f(|t_j|)$. The function f is a general membership function that takes the frequency occurrence of t_j in u as its argument and is defined by (1).

$$f(|t_i|)=\begin{cases}1 & if\ |t_i|\geq 1\\ 0 & otherwise\end{cases} \qquad (1)$$

Suppose that $NT(t_i,t_j)$ denotes that t_i is narrower than t_j. The membership degree of $NT(t_i,t_j)$ represented by $\mu_{NT}(t_i,t_j)$, is defined by (2).

$$\mu_{NT}(t_i,t_j)=\frac{\sum_{u\in C}\mu_{occur}(t_i,u)\otimes\mu_{occur}(t_j,u)}{\sum_{u\in C}\mu_{occur}(t_i,u)} \qquad (2)$$

Where $\otimes$ denotes a fuzzy conjunction operator. According to (2), the more frequent terms t_i and t_j co-occur and the less frequent term t_i occurs in documents, t_i is narrower than t_j with higher degree of confidence. A membership value of 1.0 is obtained when a term always co-occurs with another term. In contrast, the membership value of *narrower* term relation between two terms that never co-occur will be 0.

For constructing the fuzzy ontology, we should first calculate the membership values of two NT relations for each pair of two distinct terms, that is

Among the two calculated membership values for each pair of two distinct terms, the one with the higher value is selected and the other one will be omitted, hence if the two membership values are close to each other, the stronger relation will be chosen; otherwise the positive concept instance will be selected. It is obvious that less meaningful relationships should be omitted from the fuzzy ontology; it is done by applying α-cut value to the relations selected from the previous step. Finally, for each $NT(t_i,t_j)$ a search procedure is performed to see whether there is any indirect path like $NT(t_i,t_{m1}),NT(t_{m1},t_{m2}),...,NT(t_{mn},t_j)$ connecting terms t_i and t_j. If the search routine finds such a path, it is named P and the routine will calculate $\mu_{NT}(P)$ as showed in (3).

$$\mu_{NT}(p)=\min\ \{\mu_{NT}(t_q,t_k)|\ \ NT(t_q,t_k)\in P\} \qquad (3)$$

If $\mu_{NT}(t_i,t_j)$ is either smaller or equal to $\mu_{NT}(P)$, $\mu_{NT}(t_i,t_j)$ will be omitted from the fuzzy ontology.

The following illustrative example will explain the fuzzy ontology construction process in practice.

Suppose that there is a class in domain ontology which stands for "Soft Computing" concept. As mentioned before, Supervisor Agent creates a number of Search Agents; each of which relates to a specific key term. Suppose that the key term "Fuzzy Concepts" exists in "Soft Computing" class; so there will be a Search Agent which is in charge of querying the "Fuzzy Concepts" key term in order to get the most top ranked and related URLs from a keyword based search engine. We assume that the following two URLs are the result of this query:

- $U_1=www.A.com/a.html$
- $U_2=www.B.com/b.html$

```
(defrule DataBaseWebPage_4  ?instance <- (MAIN::object (is-
a WebPage) (KeyWords $? "data"  $? "information" $?
)(ifItIsProcessed_DB FALSE) )  => (slot-set ?instance
ifItIsProcessed_DB TRUE) (slot-set ?instance classType
DB_Class) (bind ?y  (str-cat
"DB_Class_StorageAgent"(System.currentTimeMillis)
":DomainSpecSearchEngine.StoreClassifiedPagesIntoDBAgent
("DB_Class"_" ?UsingQueryRefinement ")")) (call jade.Boot
main (create$ "-container" ?y))   )
```

Figure 2. A sample of web page classification rule

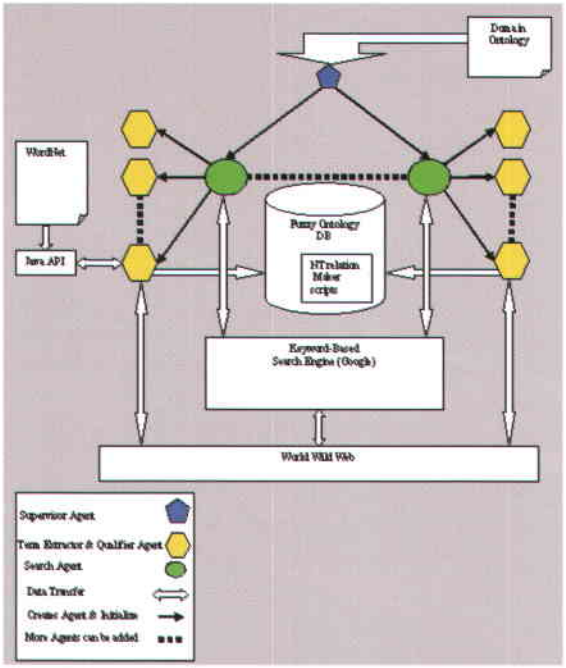

Figure 3. Proposed fuzzy ontology constructor subsystem

Since the collection of URLs will be $C=(U_1,U_2)=(www.A.com/a.html,www.B.com/b.html)$; Search Agent creates and initializes two Term Extractor & Qualifier Agents named TEQ1 and TEQ2 in order to connect to the web for retrieving the content of web pages and extracting key terms and their frequency occurrences. Suppose that validated key terms along with their frequency occurrences for each URL in C set are as illustrated in Tables I and II. In these tables, the membership value for each key term is also calculated using equation (1).

As mentioned before, the next step is calculating the membership degree for "Narrower Than" relationship between each two terms using equation (2). To clarify the way of calculating, in the following, we expanded the formula for the two key terms "Fuzzy Systems" and "Fuzzy Sets Operation". The calculation results for other pairs have shown in Table III.

$$\mu_{NT}(Fuzzy\ Systems, Fuzzy\ Sets\ Operation) = \frac{\mu_{occur}(Fuzzy\ Systems,U_1)\otimes\mu_{occur}(Fuzzy\ Sets\ Operation,U_1) + \mu_{occur}(Fuzzy\ Systems,U_2)\otimes\mu_{occur}(Fuzzy\ Sets\ Operation,U_2)}{\mu_{occur}(Fuzzy\ Systems,U_1)+\mu_{occur}(Fuzzy\ Systems,U_2)} = \frac{1\otimes1+1\otimes0}{1+1} = \frac{1}{2}$$

$$\mu_{NT}(Fuzzy\ Sets\ Operation, Fuzzy\ Systems) = \frac{\mu_{occur}(Fuzzy\ Sets\ Operation,U_1)\otimes\mu_{occur}(Fuzzy\ Systems,U_1) + \mu_{occur}(Fuzzy\ Sets\ Operation,U_2)\otimes\mu_{occur}(Fuzzy\ Systems,U_2)}{\mu_{occur}(Fuzzy\ Sets\ Operation,U_1)+\mu_{occur}(Fuzzy\ Sets\ Operation,U_2)} = \frac{1\otimes1+0\otimes1}{1+0} = 1$$

We can see that the value of $\mu_{NT}(Fuzzy\ Sets\ Operation, Fuzzy\ Systems)$ is greater than the value of $\mu_{NT}(Fuzzy\ Systems, Fuzzy\ Sets\ Operation)$, so the stronger relationship $\mu_{NT}(Fuzzy\ Sets\ Operation, Fuzzy\ Systems)$ will be chosen in order to store in database for further processes. This NT relationship shows that the "Fuzzy Sets Operation" concept is narrower than the "Fuzzy Systems" concept; which is actually true in real world. The highlighted values in Table III indicate the stronger relationships which are selected for further processes.

The relations between two distinct terms cannot be established if both terms never co-occur so that their membership values will be 0. It is obvious that unrelated terms should not be considered during the ontology creation. These terms will be automatically excluded by applying the α-cut. This is why we have no highlighted value through the 5th, 6th and 7th rows of Table III.

The information stated in Table III can be drawn as a graph as illustrated in Fig. 4. As we can see, there exist some nodes such as "Fuzzy Systems" and "Fuzzy Conjunction" which are connected to each other through more than one route (In this case the two routs are: "ab" and "c"); the excessive relations will be omitted by applying equation (3). As we can see, the result of pruning, showed in Fig. 5, reflects the actual hierarchy of fuzzy concepts in human's world. This graph will be stored in the database in order to be used for query refinement process.

TABLE I. VALIDATED KEY TERMS, THEIR CORRESPONDENT FEREQUENCY OCCURRENCES AND MEMBERSHIP VALUES FOUND IN URL U1

t_i	$occur(t_i,U_1)$	$\mu_{occur}(t_i,U_1)$
Fuzzy Systems	10	$f(\lvert t_i \rvert) = f(10) = 1$
Fuzzy Sets Operation	17	$f(\lvert t_i \rvert) = f(17) = 1$
Fuzzy Conjunction	6	$f(\lvert t_i \rvert) = f(6) = 1$

TABLE II. VALIDATED KEY TERMS, THEIR CORRESPONDENT FEREQUENCY OCCURRENCES AND MEMBERSHIP VALUES FOUND IN URL U2

t_i	$occur(t_i,U_2)$	$\mu_{occur}(t_i,U_2)$
Fuzzy Systems	12	$f(\lvert t_i \rvert) = f(12) = 1$
Fuzzy Clustering	9	$f(\lvert t_i \rvert) = f(9) = 1$
Fuzzy C-means Algorithm	15	$f(\lvert t_i \rvert) = f(15) = 1$

TABLE III. THE MEMBERSHIP DEGREE FOR "NARROWER THAN" RELATIONSHIP BETWEEN EACH PAIR OF KEY TERMS

No.	t_i	t_j	$\mu_{NT}(t_i, t_j)$	$\mu_{NT}(t_j, t_i)$
1	Fuzzy Systems	Fuzzy Conjunction	1/2	1
2	Fuzzy Systems	Fuzzy Clustering	1/2	1
3	Fuzzy Systems	Fuzzy C-means Algorithm	1/2	1
4	Fuzzy Sets Operation	Fuzzy Conjunction	1	1
5	Fuzzy Sets Operation	Fuzzy Clustering	0	0
6	Fuzzy Sets Operation	Fuzzy C-means Algorithm	0	0
7	Fuzzy Conjunction	Fuzzy Clustering	0	0
8	Fuzzy C-means Algorithm	Fuzzy Clustering	1	1

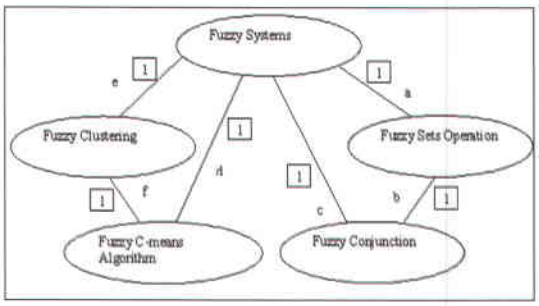

Figure 4. The fuzzy ontology with excessive relations between key terms

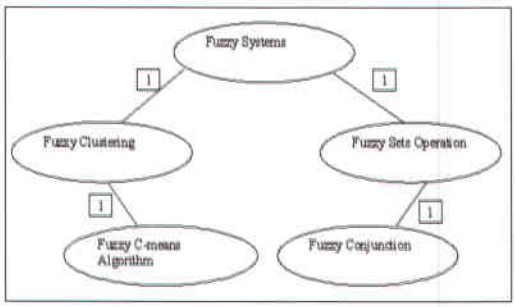

Figure 5. The fuzzy ontology of key term relations after pruning excessive relations

V. EXPERIMENTAL RESULTS

A. Implementation details

Our proposed architecture is deployed in Eclipse environment in JAVA and uses JADE[1] multi agent platform. The domain ontology has been developed by the JessTab plug-in in protégé. JESS[2] used for reasoning as the inference engine and information are stored in SQL Server 2000 relational database system.

Fig. 6 shows a snapshot of JADE platform while running the simulated environment. As mentioned before, Agents employ the Jess inference engine for their reasoning tasks, using production rules written in Jess. The general structure of such rule is:

1. Name of the rule
2. Preconditions that are required to classify a web page in a particular class.
3. Action of the rule. The presence of the preconditions in the specifications of a web page will fire the action of a rule. This action usually includes creation and initialization of a Storage Agent in order to store the specifications of a classified web page together with its correspondent class type in database.

B. The implementation domain

In simulating the proposed architecture, we defined the domain ontology on some academic concepts in field of computer which includes database, soft computing, design algorithms, parallel algorithms, software engineering, software methodologies, operating system, genetic algorithms and neural networks. Since in our proposed architecture, the Supervisor Agent creates and initializes a Search Agent for each defined concept in the domain ontology, the top level part of the domain ontology, illustrated on Fig. 7, includes a CreateSearchAgent class which maps each Search Agent to a specific concept in the domain ontology. Furthermore, the SubCategoryOf class determines the concept/sub concept relations (e.g., neural networks and genetic algorithms are two sub concepts of soft computing).

C. Parameters used for evaluation

Precision and Recall are the two parameters mostly used for evaluating the efficiency of search engines; where Precision can be seen as a measure of exactness and Recall is a measure of completeness.

Often, there is an inverse relationship between Precision and Recall, where it is possible to increase one at the cost of reducing the other. For example, an information retrieval system can often increase its Recall by retrieving more documents, at the cost of increasing number of irrelevant documents retrieved; therefore Precision and Recall scores are not discussed in isolation. Instead, both are combined into a single measure, such as F-measure, which is weighted harmonic mean of precision and recall.

Precision, Recall and F-measure are commonly evaluated as shown in (4), (5) and (6) respectively.

$$\text{Precision} = \frac{|\{\text{relevant docs}\} \cap \{\text{retrieved docs}\}|}{|\{\text{retrieved docs}\}|} \quad (4)$$

[1] Java Agent Development Framework available at: http://jade.tilab.com/

[2] Java Expert System Shell available at: http://hezberg.casandia.gov/Jess

$$\text{Recall} = \frac{|\{\text{relevant docs}\} \cap \{\text{retrieved docs}\}|}{|\{\text{relevant docs}\}|} \quad (5)$$

$$F_\beta = (1+\beta^2).\frac{\text{Precision.Recall}}{\beta^2.\text{Precision} + \text{Recall}} \quad (6)$$

For evaluating our system, we categorized search results into three groups on the basis of the following criteria and assigned each group an appropriate weight in order to obtain more precise result:

- If the web page is closely matched to the subject matter of the search query, it is categorized as "relevant" and given a score of 1.
- If the web page is somehow related to the subject matter of the search query, it is categorized as "less relevant" and given a score of 0.5.
- If the web page either is not related to the subject matter of the search query or is not accessible through the web, it is categorized as "irrelevant or inaccessible" and given a score of 0.

Thus, Precision and Recall formulas will be modified as shown in (7), (8) respectively.

$$\text{Precision} = \frac{sum\ of\ the\ scores\ of\ relevant\ \&\ retrievied\ docs}{|\{\text{retrieved docs}\}|} \quad (7)$$

$$\text{Recall} = \frac{sum\ of\ the\ scores\ of\ relevant\ \&\ retrievied\ docs}{sum\ of\ the\ scores\ of\ relevant\ docs} \quad (8)$$

We decided to use F-measure with β=1 in order to give same importance to precision and recall; that is we must retrieve all relevant documents and all retrieved documents must be relevant.

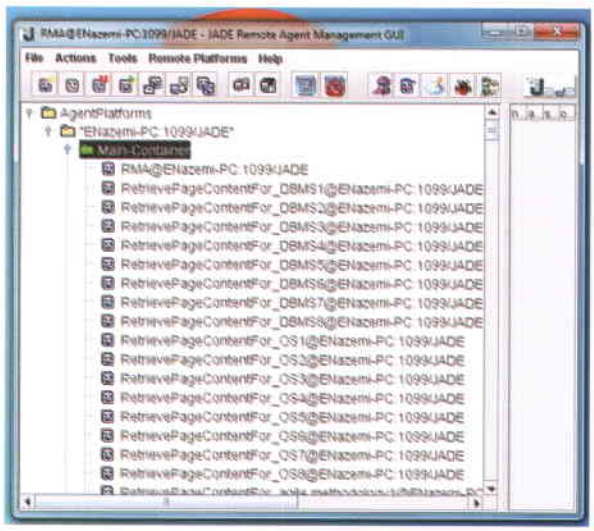

Figure 6. A snapshot of JADE platform while running the simulated environment

D. *Evaluating simulation results*

For evaluating our proposed architectural model, we launched developed system using 11 search queries as listed in Table IV.

To determine to what degree a retrieved URL is relevant to the user information needs, we asked users to see the content of each retrieved URL in the result list and assign the appropriate weight (as mentioned in previous subsection) to them. Then we calculated precision and recall for each query using equations (7) and (8).

With respect to (6) and the average values for precision and recall which are calculated in Table I, the F-measure value will be 0.956 which is actually a good result.

Fig. 8 shows a comparison between the value of evaluation parameters of our proposed architecture and the AGATHE [10] architecture.

Furthermore, in order to see the role of fuzzy ontology in improving the overall system performance, we launched our system in two different conditions listed below and then calculated the evaluation parameters.

- Search Agents consult the query refinement subsystem in order to refine their queries.
- Search Agents do not consult the query refinement subsystem.

Fig. 9 and Fig.10 show and compare the value of precision and recall in two different conditions mentioned above. We can see that the query refinement component helps Search Agents to do their task more precisely which improves the overall performance of system.

Fig. 11 also compares the average value of precision, recall and F-measure while Search Agents use and do not use the query refinement component.

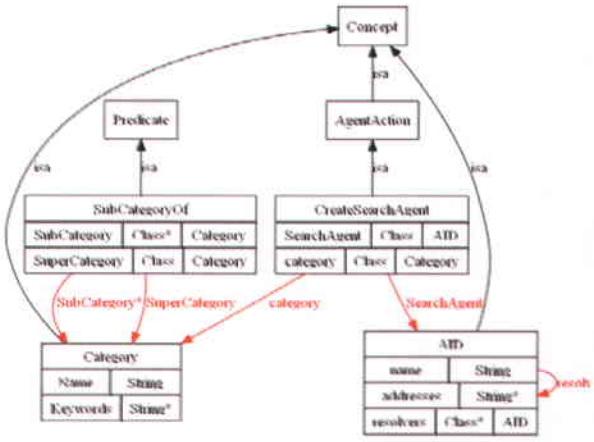

Figure 7. Top level part of domain ontology

VI. CONCLUSIONS AND FUTURE WORKS

Our proposed architecture is a multi-agent model for domain specific search engines. This architecture uses domain ontology to specify which domain is supported by the search engine. Assigning the defined tasks to intelligent agents leads to have a scalable, reliable and more efficient system.

As mentioned in previous sections, this architecture uses keyword-based search engines as its underlying layer, so for popping-up more precise results from the keyword-based search engines to upper layers, Search Agents -which have direct interaction with keyword-based search engines- use fuzzy ontology of term associations to refine their query contexts. The fuzzy ontology is made by our proposed fuzzy ontology constructor subsystem. Simulation results show that Precision and Recall are actually very good.

Since supporting end-users to tailor their queries to the underlying repository and vocabulary will help the system to present more precise results, our proposed architecture suffers from the lack of a mechanism in order to refine the input query right after it is given to the system trough the user interface. Adding an end-user query refinement component which applies the domain ontology in order to omit the potential ambiguities in users' initial queries can be considered as future work.

Another point is that the present work and its prototype support only one single domain of concepts whereas, as we know, there are some situations in which each domain can be interrelated with some other domains; for example, the tourism and academic ontologies will be interrelated when a user seeks for an international conference to attend. So our proposed architectural model for domain specific search engines can be extended to support multiple domains simultaneously.

Furthermore, since our proposed architecture for domain specific search engines depends on domain ontology, the automatic generation of domain ontology will be considered as future enhancement.

Moreover, using natural language processing techniques in Term Extractor & Qualifier Agents of fuzzy ontology constructor subsystem leads to improve their performance and can be considered as the future work.

TABLE IV. CALCULATED PRECISION AND RECALL FOR 11 SAMPLE QUERIES

No.	Query	Precision	Recall
1	Neuro Fuzzy	1	1
2	Parallel Sort Algorithm	1	1
3	Parallel Merge Algorithm	1	1
4	Shell in Operating System	0.875	1
5	What is Semaphore	0.875	1
6	Fitness	1	1
7	Software Testing	0.875	1
8	Object Oriented Methodology	1	0.8
9	Software Metrics	1	0.901
10	Linux Kernel	1	0.875
11	Chaos Theory	1	1
Average		0.965	0.961

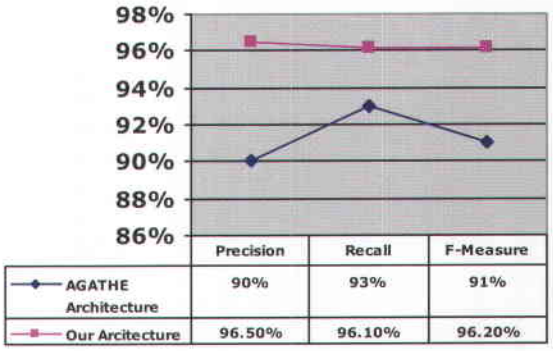

Figure 8. A comparison between the value of evaluation parameters of our proposed architecture and the AGATHE architecture.

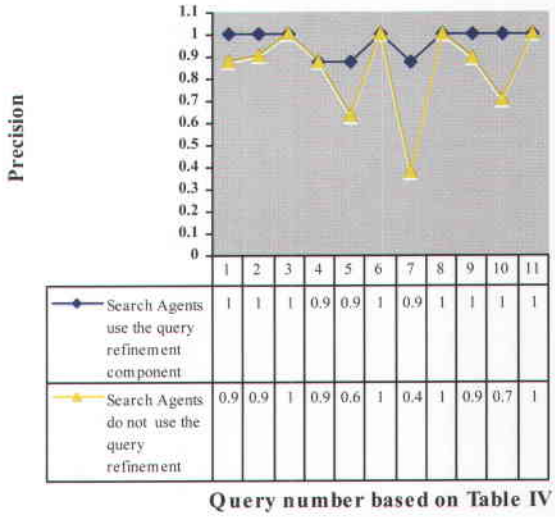

Figure 9. Comparing the value of precision while Search Agents use and do not use the query refinement component.

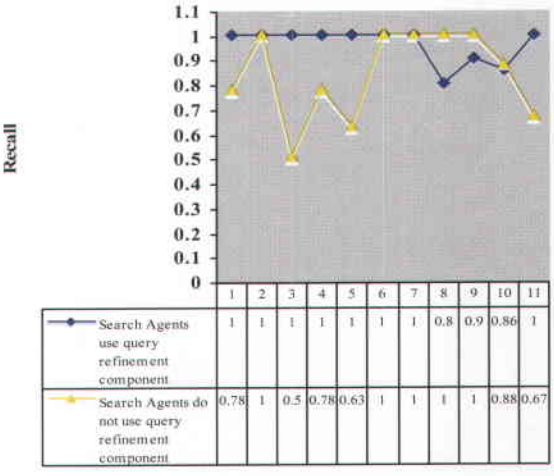

Figure 10. Comparing the value of recall while Search Agents use and do not use the query refinement component.

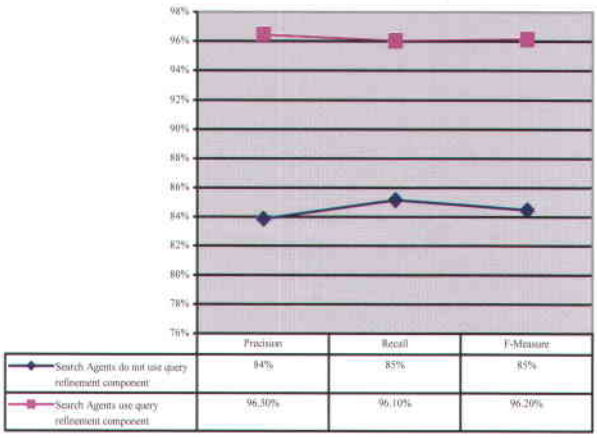

Figure 11. Compareing the average value of precision, recall and F-measure while Search Agents use and do not use the query refinement component.

REFERENCES

[1] John Davies, Richard Weeks, "QuizRDF: Search technology for the semantic web", IEEE, 2004

[2] Kyumars Sheykh Esmaili, Hassan Abolhassani, "A categorization scheme for semantic web search engines", IEEE, 2006

[3] Ben Choi, Rohit Dhawan, "Agent Space Architecture for search engines", Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'04)

[4] F. Freitas, G. Bittencourt, "An ontology-based architecture for cooperative information agents" , Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI), Acapulco, Mexico, 2003

[5] Bernard Espinasse, Sébastien Fournier, Fred Freitas, "Agent and ontology based information gathering on restricted web domains with AGATHE", ACM, March 2008.

[6] Fred Freitas, Luciano Cabral, "From MASTER-Web to AGATHE: the evolution of an architecture for manipulating information over the web using ontologies"; RECIIS – Elect. J. Commun. Inf. Innov. Health. Rio de Janeiro, v.2, n.1, p.73-84, Jan.-Jun., 2008

[7] Dwi H.Widyantoro, John Yen, " A fuzzy ontology-based abstract search engine and its user studies", Department of Computer Sciences Texas A&M University. IEEE (2001) , p. 1291-1294

[8] Han Lixin, Chen Guihai, "HQE: A hybrid method for query expansion", Elsevier Ltd., 2008

[9] Haytham T. Al-Feel, Magdy Koutb, Hoda Suoror, "Semantic web on scope: a new architectural model for the semantic web", Journal of Computer Science 4 (7): 613-624, 2008

[10] Bernard Espinasse, Sébastien Fournier, Fred Freitas, "AGATHE : An agent and ontology-based system for gathering information about restricted web domains", International Journal of E-Business Research, 5(3), 14-34, July-September 2009

[11] S. M. BaalaMithra, S. M. SominMithraa, "Context-aware Automatic Query Refinement Using Indian-Logic Based Ontology", Third International Conference on Advances in Semantic Processing, IEEE, 2009

[12] Masayuki Okabe, Seiji Yamada, "Semisupervised Query Expansion with Minimal Feedback", IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. 19, NO. 11, NOVEMBER 2007

[13] Liangjun Ma, Lin Chen, Yibo Gao and Yiping Yang, "Ontology based Query Expansion in Vertical Search Engine", Sixth International Conference on Fuzzy Systems and Knowledge Discovery, IEEE, 2009

[14] Fardin Akhlaghian, Batool Arzanian, Parham Moradi, "A Personalized Search Engine Using Ontology-based Fuzzy Concept Networks", International Conference on Data Storage and Data Engineering, IEEE, 2010

Elmira Hajimani was born in Tehran, Iran in 1984. She received her B.Sc. degree in Software Engineering from Shahid Beheshti University, Tehran, Iran in 2007, and the M.Sc. degree in Software Engineering from Science and Research Branch, Islamic Azad University, Tehran, Iran in 2010. Her current research interests include Semantic based Search Engines, Web Mining and Information Retrieval Systems.

Eslam Nazemi was born in Sarab, Iran, in 1954. He received the B.Sc. degree in Applied Maths & Operational Research from School of Planning and Computer Application, Tehran, Iran in 1977, the M.Sc. degree in Both System Engineering and Economics in 1987 and 1996, and PhD. in Information Technology in 2005. He was the faculty member from 1978 in School of Planning & Computer Application and then from 1986 to the preset, he has been with the Electrical & Computer engineering Faculty at Shahid Beheshti University, Tehran, Iran. He is now an Assistant Professor and the deputy of graduate and education affairs. His main fields of research are Software Engineering, Large Scale Software Development, Search engines, Web Mining, and Software Quality. He has authored and coauthored more the 50 papers and has 10 books on maths, project management, software eng. and game theory.