A Distributed Agent Based Web Search using a Genetic Algorithm

M. Koorangi, K. Zamanifar

Department of Computer, Faculty of Engineering, University of Isfahan , Iran

Summary
In this paper, the problems of current web search engines are analyzed, and the need for a new design is justified. Some ideas on how to improve current web search engines are presented, and then an adaptive method for web meta-search engines with a multi-agent specially the mobile agents is presented to make search engines work more efficiently. In the method, the cooperation between stationary and mobile agents is used to make more efficiency. The meta-search engine gives the user needed documents based on the multi-stage mechanism. The merge of the results obtained from the search engines in the network is done in parallel. Using a reduction parallel algorithm, the efficiency of this method is increased. Furthermore, a feedback mechanism gives the meta-search engine the user's suggestions about the found documents, which leads to a new query using a genetic algorithm. In the new search stage, more relevant documents are given to the user. The practical experiments were performed in Aglets programming environment. The results achieved from these experiments confirm the efficiency and adaptability of the method.

Key words:
Mobile Agent, Meta-Search Engine, parallel, Distributed, Genetic Algorithm

1. Introduction

WEB has became a default to locate information and the ideal tool to find such information is search engine[2]. Search engine as a tool to investigate the web must obtain the desired results for any given query. Success of a search engine is directly dependent on the satisfaction level of the user. Users desire the information to be presented to them within a short time interval. They also expect that the most relevant and recent information to be presented [2]. Most of the search engines cannot completely satisfy user's requirements and the search results are often very inaccurate and irrelevant [5].

There are already many researchers who have reported on about various aspects of search engines in [18 through 31,47,51]. Search tools for the web can be classified as Search Engines, Directory Services, Meta-¬Search Engines, and Hybrid Search Services. Google, Inktomi, etc., are typical search engines. Yahoo is a popular directory service. Meta-Crawler, ProFusion, Vivisimo etc., are meta-search engines. Msn search and Yahoo can be termed as hybrid search services too, because they have a search engine as well as directory services incorporated in them. Each search engine has three key functional phases, namely, Web Data Acquisition (WDA), Web Data Indexing (WDI) and Web Data Rendering (WDR).

There are already many researchers who have reported on about various aspects of search engines in [18 through 31,47,51,]. Search tools for the web can be classified as Search Engines, Directory Services, Meta-¬Search Engines, and Hybrid Search Services. Google, Inktomi, etc., are typical search engines. Yahoo is a popular directory service. Meta-Crawler, ProFusion, Vivisimo etc., are meta-search engines. Msn search and Yahoo can be termed as hybrid search services too, because they have a search engine as well as directory services incorporated in them. Each search engine has three key functional phases, namely, Web Data Acquisition (WDA), Web Data Indexing (WDI) and Web Data Rendering (WDR). They are divided to general purpose and special purpose [2]. A meta- search engine is the kind of search engine that does not have its own database of web pages. It sends search terms to the databases maintained by other search engines and gives users the results that come from all the search engines queried [5].

1.1 Problems and Limitations

The problems and limitations of the search engines include:

1) Maintaining the freshness with respect to the change frequency of the web is a gargantuan task [2]. In the current information age, the web is increasing at a very rapid pace, while the indexing of the current search engines is not scaling up at the same pace resulting in the loss of access to good function of documents on the web. Current technology is inadequate in indexing the entire web [8].
2) Consumption of huge bandwidth [2].
3) Crawlers consume majority of web server time [2,48].
4) The resources can occur many times due to mirroring and aliasing [3].
5) There are several limitations using web crawlers to collect data for search engines: Not Scalable, Slow Update, Hidden (Deep) Web, Robot Exclusion Rule, High Maintenance [34].

Manuscript received January 5, 2007
Manuscript revised January 25, 2007

Intelligent Agent	Agent	Intelligent	Software Engineering	Engineering	Software	Genetic Algorithm	Genetic	Algorithm	
48800000	83000000	55600000	87500000	321000000	886000000	1500000	40900000	27400000	Google
1230000	52230475	19327552	27145230	528531455	384331700	375241	8841535	4697822	Msn
9400000	350000000	123000000	102000000	339000000	1720000000	2430000	62100000	27200000	Yahoo

Fig. 1 Number of results for different words in Google, Msn and Yahoo.

6) A successful search engine system requires a large data cache with tens of thousands of processors to inverted text indices, to measure page quality, and to execute user queries [35].
7) Centralized systems provide a single point of failure. Failures may be network outages; denial-of-service attacks; or censorship by domestic of foreign authorities [1].
8) Client/server networks architectures because of focus on the server, provide a bottleneck, therefore they are not fault tolerant. They also have complex architecture and delay in remote networks [31,46].
9) Information overlap: During the study of this research, searching different words within various search engines has provided a lot of results. Surely most of them don't fit the user's real requests. The figure1 refers to the conclusion.
10) Heterogeneous and distributed information [40,43,45].

In such circumstances, it is very necessary to be scalable systems, so that as the network is extended, it will be possible to access updated information by an acceptable expense.

2. Related Works

Finding information on the Internet web search engines, like Google, Yahoo and Altavista, is one of the top three Internet activities according to reference [9]. Though meta-search engines address some of the main drawbacks of the search engines, they may sometimes result poor precision brought by the heterogeneity of the underlying search engines. In other words, the query that can be used to optimally describe a user's particular information need may vary from one search engine to another [8]. Many researchers have been working on meta-search engines[8,11,12,13,14,15,16,17,34,35,36,37,41,49,50].
Result of researches show that a distributed system is more reliable than a centralized system. Also work load in a centralized search engine would concentrate on a few hosts and it is impossible for the hosts to cooperate with final users [1].
Web search engines have traditionally relied on enumerating the entire web using crawlers, with results in either lag or inefficiency if the frequency of crawling differs from the frequency of updates for a given page [1,42,44]. A mechanism called "push" can be used to improve the efficiency of search engine in peer-to-peer networks. It means that the clients give the new contents to the servers directly. Although the push mechanism can be used, crawlers are used in the web normally. The expectation of clients to update the available information in the web search engines is impossible because of variety of clients and servers and heterogeneous information and the lack of control them.
Indexing is divided by two partitioning schemes: horizontal and vertical. A horizontally partitioned index divides this list among several nodes, either sequentially or by partitioning the document identifier space. A vertically partitioned index assigns each keyword, undivided, to a single node. Therefore, in vertical scheme, queries are sent to the fixed number of hosts, while in horizontal scheme, queries are broadcasted in all nodes. Thus, the throughput of a vertically partitioned index theoretically grows linearly as more nodes are added. Query throughput in a horizontally partitioned does not benefit at all from additional nodes (for more information refer to references [1, 4]).
In reference [1], a distributed method based on Vertical indexes has been proposed, that each query is sent only to its relevant node, and then final result is identified by combining the different server results. Bloom filter and data cache are used to improve this method.
In this method it can be said that the search cost will be increased because of the k server participation in answering the k keyword query. But this method can not be used for search engines, because the search engines index the documents by crawling and then indexing cost is high. In the other hand, connecting, disconnecting and changing the nodes lead to update their information, and it is very expensive.
In reference [2], regarding to the real time issues and also the generic architecture of search engines, a prototype based on a mesh has been presented. Web data is distributed onto various processors of the parallel system. The number of nodes that must be processed in each processor are calculated by , where h is number of levels in the N-ary tree and k is the number of processors in the parallel system. It can be said that this method is presented only in theory, and it's implementation has high cost.
Reference [3] has presented an adaptive web search system based on a multi-agent reactive architecture, which comes from biological researches on the ant searching behavior [10]. The algorithm has proven to be robust against environment alternations and adaptive to user's

information need changes, discovering valuable evaluation results from standard web collections. A large number of agents are trying to satisfy user's requests. Collectively this method believes that "If a page relates to user's favorite topic, and if he is currently visiting this page, he will probably look at the linked pages, because they could be probably related to the topic at issue". In this method there are crawlers for each user search which study available links on behalf of the user.
It can be said that the direction of the method is mostly toward special purpose applications. Also because of departing a lot of agents through the web for each query, if This method is still in research mode and can be a basis of future meta-search engines. There are a few critics of this method such as: broadcasting the query to the whole underlying search engines, the possibility of making bottleneck on the side of top agent while the results return, hard and time consuming task of top agent to merge results, and the possibility of making heavy traffic.

3. The presented search method

In this section, an adaptive method based on multi-agent is presented to design a meta-search engine. This method uses a distributed architecture of stationary and mobile agents through the web. The primary idea of this research was that the coordinator agent make several mobile agents, and they move to different sites simultaneously and submit the retrieved information to the coordinator agent in order to be given to the user, after merging them. The primary design was not completely efficient and was gradually improved. At first the primary design and then the improved ones are presented. Also more improvements can be c onsidered in section 6.

it will get applied as a normal search engine, its efficiency is reduced.
Reference [8] has presented an adaptive method to improve the cure of the search results. This method uses a neural network model on agent to design an adaptive meta-search engine in order for it to improve search results by computing user's relevance feedback. The proposed model indicates a multiple layer neural network relating to the user's specific information need. If the user provides certain relevance feedback, this neural network can adjust the weight for each search engine.

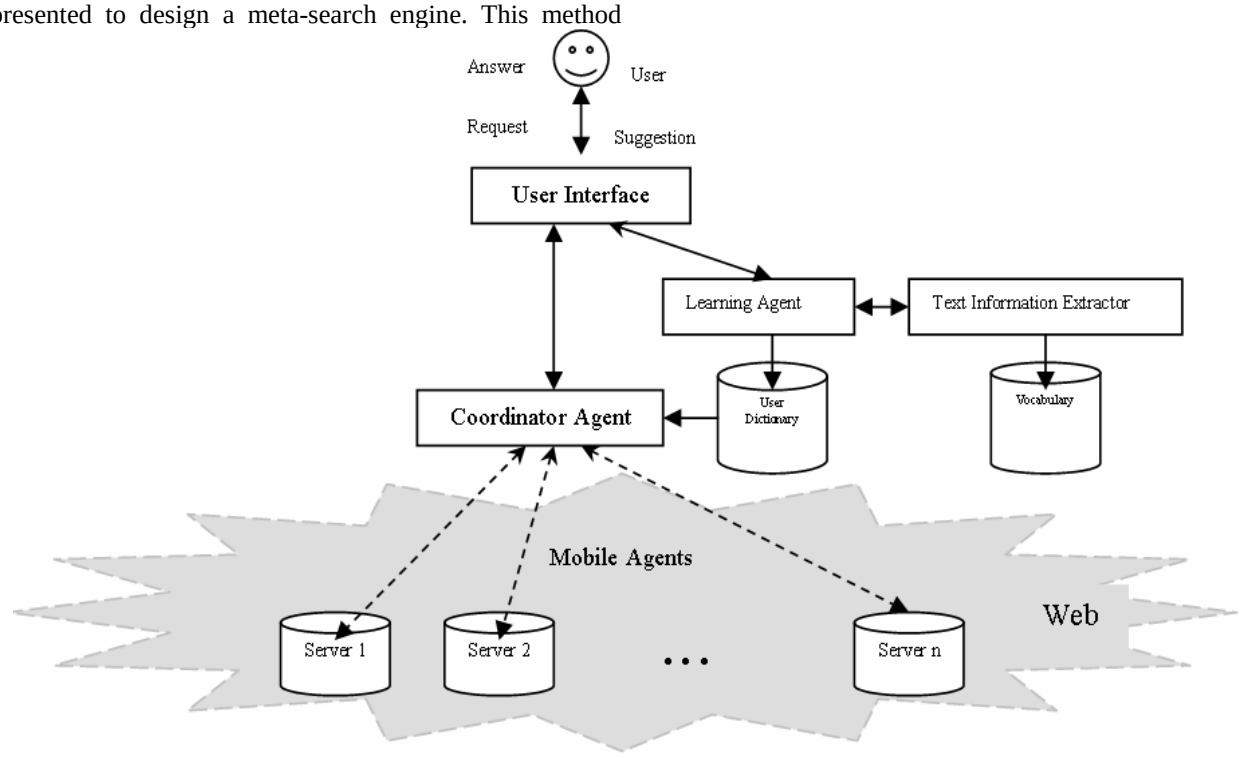

Fig. 2 The primary method for search

3-1. The primary method: A distributed adaptive search in the web

In this primary design, the search was only based on the mobile agents. Figure2 shows this method .
Based on this method as shown figure 2, the local servers index their information locally. A coordinator agent first gets the user request and creates a query, and then creates some of the mobile agents and sends them to the web servers in parallel. Each mobile agent then moves to the relevant server, and performs high bandwidth local index searching. Then the mobile agents comeback to the coordinator agent and submit the retrieved information. Then the coordinator agent merges the indexed lists. Then they are given to the user by the user interface. After reviewing the documents, the user submits his suggestions to the user interface by words such as: excellent, very good, good, pretty good, bad and very bad. The marks such as 3, 2, 1, 0, -1, -2 , … are allocated to each page. The learning agent is responsible to learning the user's interests. To do this, the learning agent uses the text information extractor. Briefly speaking, the text information extractor analyzes the pages, removes the HTML tags, scripts commands and unnecessary words. The learning agent displays the user dictionary to the user. The user changes the weight of keywords arbitrary. The user dictionary includes a two-columned table. The columns are related to keyword and its weight, respectively. The weighty keywords are placed in the top of lists. Therefore, in this research, the weights of keywords are calculated by three factors: 1) The marks which is given to the pages by the user (S_1). 2) The keyword repetition in page (tf_P). 3) The mark which is given to keyword by the user (S_2). Above subjects are summarized in following formula: $tf_D = tf_D + \frac{S1}{100} * \frac{S2}{100} * tf_P$ (where tf_D is frequency of the keyword in the user dictionary). The coordinator agent selects the population of the keywords randomly and then creates the new query using the logical operators. This query is given to the mobile agent for new searching (more information on the genetic algorithm of the searching information can be seen in the references [6, 7, 39]). This process continues until the user finds the needed information. In fact, meta-search engine adapts itself with the main user's information need step by step. However the problems which exist in this method are:

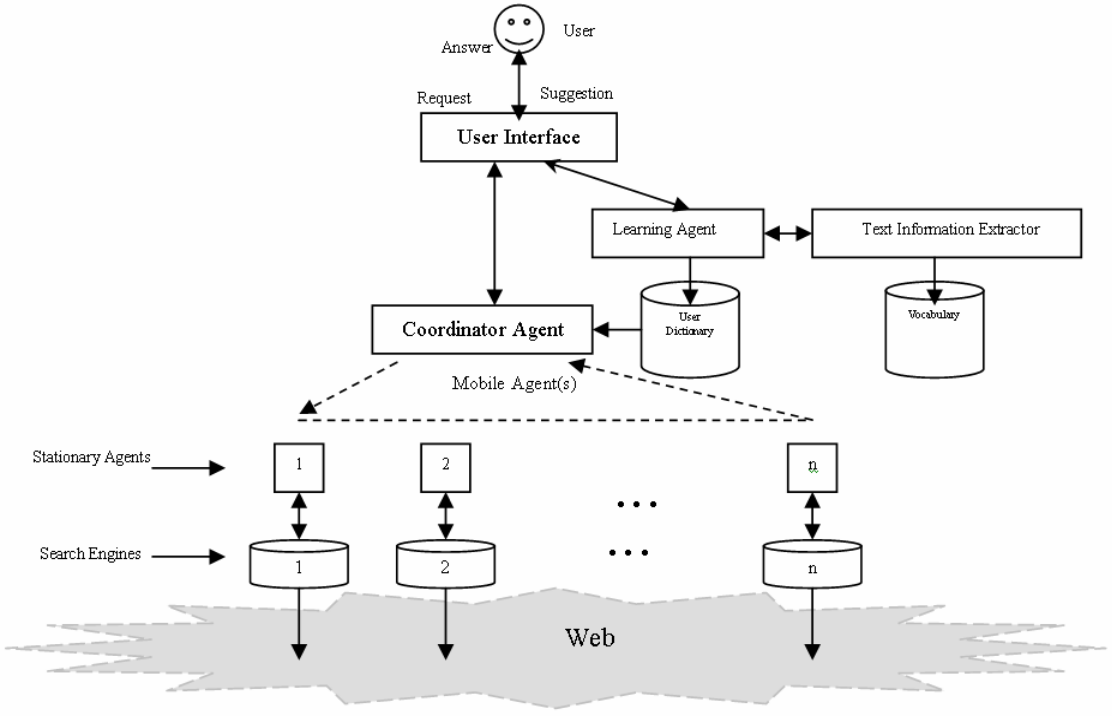

Fig. 3 The first proposed method for meta-search engine

1) Regarding to the big number of sites, if an agent is made and sent for each one, the number of the mobile agents will increase and it cause a big traffic in the network.
2) Regarding to the big number of sites, if only few agents are made and each one is responsible of searching on some sites, it cause the search will face with much delay.
3) Due to possible creation of the bottleneck on the side of the coordinator agent, it may be locked.
4) There is no use of the search engines which are active in the web and are constantly indexing the content of the web.
5) The web servers have to index their information.

Regarding to the problems in the primary method, the following improvement methods are presented which seems more suitable for the web environment.

3-2. First improved method : A distributed and adaptive meta-search engine

The first improved method is shown in figure3.
In this method, multiple active search engines in the web are applied. If the search engines are the special purposes, it leads to increase more searching and decrease repeated information. Here, each stationary agent handles the communication details of one search engine and is responsible for connecting and receiving search results from it. According to the number of stationary agents, the coordinator agent submits the weighted query to them by using one or more mobile agents. The number of the mobile agents can vary depending on the situation of the network. It is the duty of the mobile agent(s) to move and inform the stationary ones about the query and the result merge map. Notice that the number of the mobile agents is less than the number of the search engines. Sending one or more mobile agents can be done for several reasons :

- Avoiding the issue of the broadcasting command in the network.
- Determining the map of merging dynamically.
- Possibility of the determination of different queries for different search engines.
- Possibility of deleting the search engines in the next queries.
- Possibility of adding more new search engines in the next queries.

The stationary agents then dispatch query terms independently to the corresponding search engines. Then the results of these queries are merged after returning. Here, two important points should be considered. Each stationary agent uses two factors to ranks the indexes. One, the ranks presented by the search engine, and the other the weight of each query term which come from the coordinator agent. First the value of each document d_i (based on each query term q_k) is calculated by $v_{i,k} = (N + 1) - P / N$ in which N is total number of returned documents by search engine, and P is the position of document d_i in the result set. Then, for each document d_i , the local value is calculated by: $v_{i,local} = \sum_{k \in Q} v_{i,k} \times w_k$ Where w_k is weight of q_k . Since each query term has its own weight in this formula, similarity measure between query and document is calculated according to both query terms and documents keywords. So the fitness of document is calculated by both user relevant feedback factor and rank of relevant search engine.

Ever since, the stationary agents provided the ranked lists of indexes. These ranked lists are contained many items. If all of them will be given to the user, the efficiency of the search will be decreased because of many reasons such as:

- Irrelevancy of these documents and the user's information need.
- Creation of high traffic by transferring them in the network.

As a result, N tops of ranked lists of indexes are returned to the user. Users rarely need all the search results of a keyword search. Therefore, part of the desired results could be returned to them. Consequently the need to send the data will be reduced. This is vital for the scalability of the method. This way is effective to the achievement of constant cost independently of the number of the network documents. Processing the results merge is described in section 4.

3-3. Second improved method : A distributed and adaptive meta-search engine

The second improved method is shown in figure4.
In this method, also multiple active search engines in the web are applied. About difference of this method with an improved one, this method believes that first the coordinator agent will recognize the sites which are interested of the user by using of underlying search engines. Then in the next queries, it gives query to both the search engines and the mobile agents. Each mobile agent moves to some of the sites, in order to investigate the information of them using high bandwidth and to return the considered information. This improvement come from this idea that in the initial queries, if a site is considered as a favored one for the user, then in the next steps will be considered more. This method leads to access to the more updated information, because there is the access to the

information indexes through the search engines, and to the information

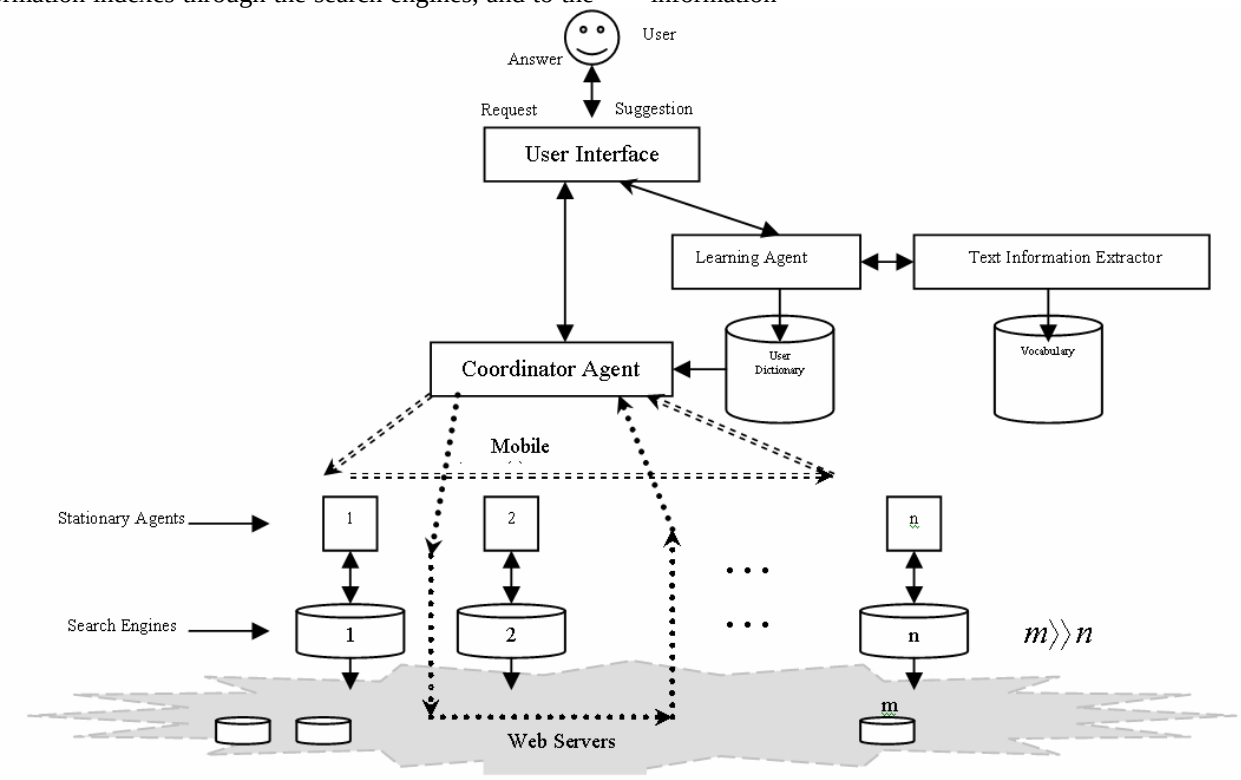

Fig. 4 The second proposed method for meta-search engine

through the mobile agents. The fitness of document has been considered according to three factors: 1) Search Engine Ranks 2) User Relevant Feedback 3) Weights of the keywords in the documents. So the relation between the user's needs and the acquired documents will be increased. Of course, for performing of this method, each server needs to the learning agent, the information text extractor and the vocabulary.

4. Results merge algorithm

When the stationary agents provide their ranked lists, they don't send them to the coordinator agent directly. Else a bottleneck is created on the side of the coordinator agent, which leads to the decreasing of meta-search engine efficiency. So, by using a parallel reduction algorithm that is executed in the network, the information is merged and N tops of ranked lists are returned to the coordinator agent. Regarding to figure5 and the pseudo code in algorithm1, the results merge algorithm is executed. All the stationary agents execute the algorithm in parallel.

To insert "Tables" or "Figures", please paste the data as stated below. All tables and figures must be given sequential numbers (1, 2, 3, etc.) and have a caption placed below the figure ("FigCaption") or above the table("FigTalbe") being described, using 8pt font and please make use of the specified style "caption" from the drop-down menu of style categories.

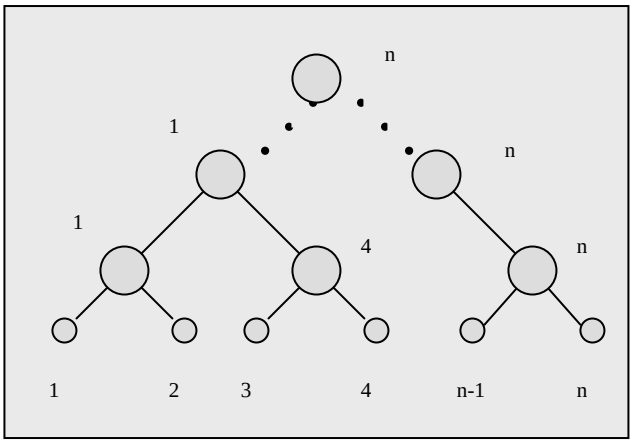

Fig. 5 The binary tree of Parallel Reduction Algorithm

```
1  Algorithm to Merge of Search Results from n Server Using n Agent
2  Initial Condition : List of n Server
3  Global Variables : n , i , j
4  Begin
5   For  j ←—— 0  to  ⌈log n⌉  Do
6      { If ( i  mod  2^j  <> 0 ) Then
7          {  Send Results to  StationaryAgent_{i+2^j−j} ;
8             Finish ;
9          }
10       Else
11         {  Receive Results from  StationaryAgent_i ;
12            Merge Results ;
13         }
14     }
15  Send Results to CoordinatorAgent ;
16 End
```

Algorithm1. The results merge algorithm

Each stationary agent executes the algorithm at least 1 time, and at most $\log n$ times. The stationary agent calculates the value of $k = i \bmod 2^j$. If value is anything except zero, it is clear that is a sender, and if value is equal to zero, the agent is a receiver. In row 7 of the algorithm, the sender agent sends its information to the receiver agent, and in row 8, terminates. In row 11, the receiver agent receives the information from the sender agent, and in row 12, merges the results.

4.1 Example of the results merge algorithm

For distinguishing the algorithm mechanism, consider an example of combination of 8 search engines. Each agent at least 1 time and at most 3 times participates in the merge process. Thus the variable j in each agent at least gets value 1 and at most gets value 3. For j=1, value of $k = i \bmod 2^j$ is calculated by the various agents as follows:

8	7	6	5	4	3	2	1	i
0	1	0	1	0	1	0	1	k

It is clear that agents 1, 3, 5 and 7 are senders and agents 2, 4, 6 and 8 are receivers. After performing step 1, agents 1, 3, 5 and 7 are excluded from the merge process, and the remaining agents that are four, 2, 4, 6 and 8, continue the merge process. Thus, this continues with j=2.

8	6	4	2	i
0	2	0	2	k

Agents 2 and 6 are senders and agents 4 and 8 are receivers. And in step 3:

8	4	i
0	4	k

Agent 4 is the sender and agent 8 is the receiver. In the next step, only agent 8 is remained that executes row 15 of the algorithm, which sends the results to the coordinator agents.

5. Experimental Results

5.1 Experimental consequences of results merge algorithm

Some examples of mobile objects in the web are Java apples, Servlet and Aglets. Java applets are program codes that can be downloaded, instantiated, and executed. The servlet allows the client to upload additional program code to a server. The servlet's code is then instantiated and executed in the server. Aglets are Java objects that can move from one host on the Internet to another. That is, an aglet that executes on one host can suddenly halt execution, dispatch to a remote host, and resume execution there. When the aglet moves, it brings its program code as well as its data [32, 33, 38].

Implementation and execution of the results merge algorithm were performed on Aglets agent environment. Aglet is an environment for the designing and execution of mobile and stationary software agents which is designed by IBM and includes object-oriented programming. Also in Aglets environment, some mechanisms have been provided for security of the information. Aglets object, can continue by local and remote massage passing [32,33,38].

The program was experimented on 3, 5, 9, 17 computers in a local network, and the following results were acquired. Regarding the change of network situation in different times, each experiment repeated 30 times. The amount of the starting and the ending of activities were measured and were recorded by the program itself. Since in each of these experiments, a computer is regarded as an origin (tree's root), the results can be only based on number of searched computers. As can be seen in figure6, a comparison is made on the average time of the merge program execution between serial and parallel on 2, 4, 8 and 16 computers. The above line of the figure6 shows the time of serial execution of program, and the low line shows the time of parallel execution of program. According to the figure6, if the number of computers be increased, the difference of the execution time between serial and parallel programs will be increased. The difference of the times can be important for the meta-search engine that has many underlying search engines. Figure7 shows the speedup of parallel execution related to serial execution.

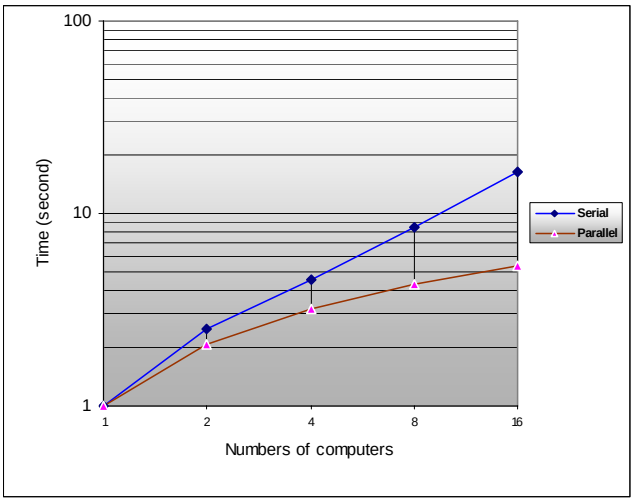

Fig. 6 The execution of parallel program in comparison with serial

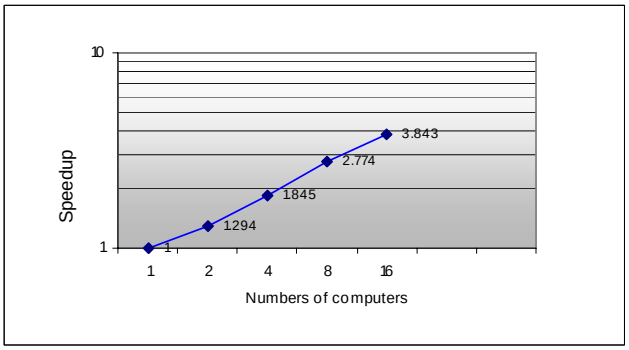

Fig. 7 Speedup of parallel in comparison with serial of program execution

The experiments confirmed theoretical results of the program. Regarding the acquired results, time of algorithm execution was $t \log n$ (where n is the number of computers and t is the basic time with two computers).

5.2. Experimental results of the method adaptability

As an initial experiment of using the program, a user who is supposed to access the last information about the computer science was considered. The user tends to know about software, and starts with keywords "Computer" and "Research". In this experiment, two search engines, "Google" and "Msn" were used, and only PDF pages was retrieved and the pages which each time were submitted to the user were 3. The first 5 keywords of the user dictionary have been used as genetic population to select the query keywords. Each time, 3 keywords through that population have been selected randomly for creation of query. The abstract of the papers or at most 20 lines of the beginning of the papers were selected to extract keywords. By each time of presenting the results to the user and having seen its feedback, the keywords of the dictionary were more updated. Also the final opinion about priority of the keywords is applied by the user. The summary of the experimental results can be seen in figures 8 through 14. Figure8 shows the form of acquired keywords with tf_P , $S1$, $S2$, tf_D in Java programming environment. Figure9 shows the acquired articles and percentage of relevance them to user's information need in the first stage of search. Figure10 shows the keywords and their ranks acquired from the articles of figure9. Figure11 shows the five keywords of top of list of user dictionary selected from figure10. Figure12 shows Status of Agents 1 through 3. Figure13 shows summary of six sequential queries, acquired articles, keywords added to user dictionary and percentage of relevance to user's information need. Figure14 shows the percentage of relevance of the retrieved documents with the user's information need.

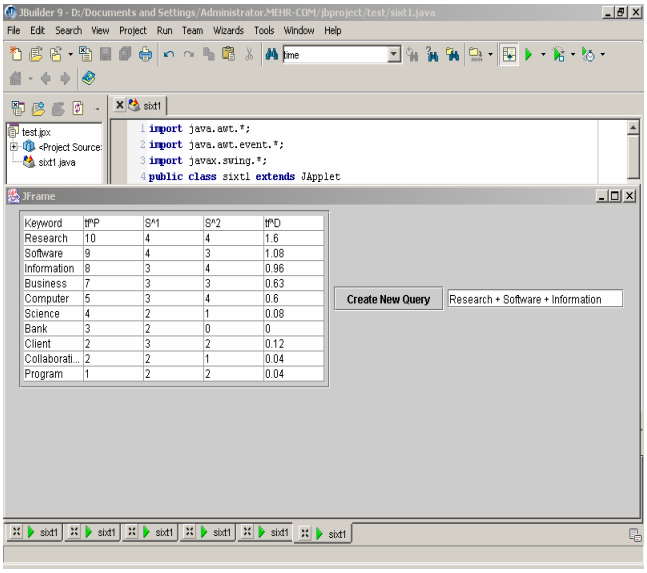

Fig. 8 The form of acquired keywords with tf_P , $S1$, $S2$, tf_D

Percentage of relevance to user's information need	Acquired articles	
%5	Computer Research, Inc. Implement Premier Business ...	1
%5	Application for Individual Membership Institute …	2
%5	Apply to Ph.D. Programs in Computer Science 2003 …	3

Fig. 9 The acquired articles in the first stage of search

tf_D	$\frac{S_1}{10} \times \frac{S_2}{10} \times tf_P$	S_2	S_1	tf_P	**Keywords**
1.6	1.6	4	4	10	Research
1.08	1.08	3	4	9	Software
0.96	0.96	4	3	8	Information
0.63	0.63	3	3	7	Business
0.6	0.6	4	3	5	Computer
0.08	0.08	1	2	4	Science
0	0	0	2	3	Bank
0.12	0.12	2	3	2	Client
0.04	0.04	1	2	2	Collaboration
0.04	0.04	2	2	1	Program

Fig. 10 Keywords and their ranks acquired from the articles of figure9

tf_D	**Keywords**
1.6	Research
1.08	Software
0.96	Information
0.63	Business
0.6	Computer

Fig. 11 Five keywords of top of list of user dictionary

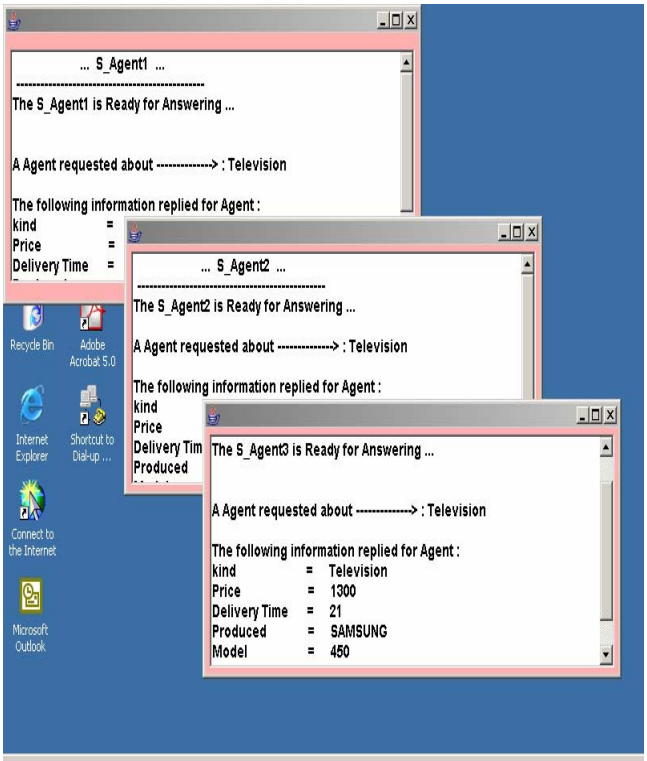

Fig. 12 Status of Agents 1 through 3

6. Evaluation and suggestion of future activities

In the method presented in this paper using the cooperation between stationary and mobile agents, it was tried to reach a higher efficiency in comparison with the former method. The specifications of proposed method are:

1) The lack of issuing broadcast command in the network from the coordinator agent.
2) The lack of making bottleneck on the side of the coordinator agent, when results are returned from the search engines.
3) Using parallel reduction algorithm, merging results is executed in parallel and with more speed.
4) In the network, smaller lists are exchanged among search engines, and therefore, it leads to decrease the traffic in the network.
5) Searching and merging results are divided between agents, and load balancing is done properly.
6) The computing power of all network nodes is extremely utilized.
7) By returning only the predetermined number of results to the user in each step, the bandwidth cost of returning the results is constant and independent of the number of nodes in the network. It confirms the scalability of the method.

With the lack of issuing broadcast command, high traffic in network can be avoided, but sending query to stationary agents may be performed with lower speed. In this case, the advantage of the lack of high traffic has been preferred to the speed of sending queries. In relation to the future activity of this method, several important points can be taken into consideration:

1) The first one is to send the population of keywords instead of sending a query to the stationary agents. The stationary agents after receiving keywords using the genetic algorithm create queries and submit to the search engines autonomously. Making these new queries and submitting them to search engines continues on to reaching predetermined fitness.
2) The second improvement is that, when each mobile agent observed a page with an upper fitness comparing to the found fitness up to that time, it should send a message to other mobile agents immediately. Then they set their movements based on the messages until the final better information acquired.

Percentage of relevance to user's information need	5 keywords of top of the dictionary	Keywords added to dictionary	Acquired articles	New queries	
% 5	Research, Software, Information, Business, Computer	Research, Software, Information, Business, Computer, Science, Bank, Client, Collaboration, Program	1-Computer Research, Inc. Implement Premier Business 2-Application for Individual Membership Institute 3-Apply to Ph.D. Programs in Computer Science 2003	Research+ Computer	1
% 16	Business, Research, Knowledge, Market, Software	Knowledge, Market, Mobile, Intelligent, Agent, Commercial, Technology	1-The most Comprehensive Internet Research Solution 2-Research Seminar, Information, Strategy, Systems … 3-OULU Advanced Research on Software, Information	Research+ Software + Information	2
% 35	Knowledge, Management, Intelligent, Market, Research	Management, Flexibility, Distribution, e-Knowledge, Cooperative, Portal	1-Managing Corporate Knowledge to Create Strategic… 2-The e-Knowledge Based Innovation Seminar 3-Making a Market in Knowledge …	Business+ Knowledge + Market	3
% 40	Artificial-Intelligent, Knowledge-Management, Semantic-Web	Knowledge-Generating, Knowledge-Oriented, Semantic-Web	1-Knowledge Management ... 2-Ontology-based user modeling for knowledge manage ... 3-A case of intelligent analysis and knowledge	Intelligent+ Knowledge+ Management	4
% 55	Learning, Multi-Agent, Knowledge-Intelligent, Artificial-Intelligent	Learning, Knowledge-Engineering, Collaborative-Learning, Personal-Agent, Software-Agent, Learning-Process, Multi-agent, Interaction, ...	1-The methodology of collaborative synthesis by … 2-Intelligent Directors for Argumentative learning … 3-Collaborative and competitive Decision Making	Artificial+ Intelligent+ Collaboration	5
% 75			1-Learning in Multi-Agent Systems … 2-Special Session on Multi-Agent Learning … 3-Multi-Agent Systems …	Multi-Agent+ Learning	6

Fig. 13 Summary of Six sequential queries and their results

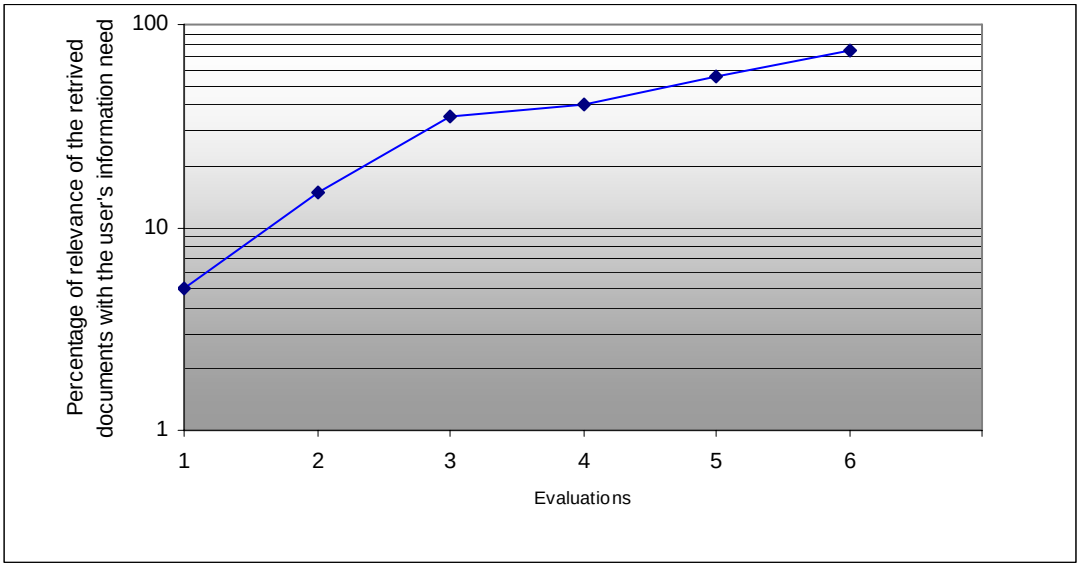

Fig. 14 Percentage of relevance of the retrieved documents with the user's information need

7. Conclusion

In this article, the current problem of the search engines and meta-search engines were studied. Merits and demits each was explained in brief and then a search method was presented where it tried to achieve a high efficiency using the cooperation of mobile and stationary agents. In this method, enhancing the efficiency of a meta-search engine was considered by removing network problems especially network traffic and bottleneck. Also by presenting a parallel reduction algorithm for merging the results of search engines, more efforts were made to speed up merging and utilizing different network nodes. The achieved results of the practical experiments confirm the efficiency and the adaptability of the proposed method. Our theoretical understand was realized for improving the method.

References

[1] P. Reynolds, A. Vahdat, "Efficient Peer-to-Peer Keyword Searching", Department of Computer Science, Duke University, Supported by Hewlett Packard, IBM, Intel, and Microsoft, 2002.

[2] K.Satya Sai Prakash, S. V. Raghavan, "DIAPANGSE : Distributed Intelligent Agent based Parallel Architecture for Next Generation Search Engines", Dept. of Computer Science & Engineering, Indian Institute of Technology Madras, India, 2001.

[3] F. Gasparetti, A. Micarelli, "Swarm Intelligence : Agents for Adaptive Web Search", Dept. of Information, University of ROMA TRE, Rome, Italy, 2000.

[4] T. Seul, C. Mathur, Jo-Wen Wu, J. Zhang, A. Delis, M. Kharrazi, X. long, K. Shanmugasundaram, "ODISSEA : A Peer-to-Peer Architecture for Scalable Web Search and Information Retrieval", Dept. of Computer and Information Science, Polytechnic University, 2003.

[5] Z. Li, Y. Wang, V. Oria, "A New Architecture for Web Meta-Search Engines", Seventh Americas Conference on Information Systems, CIS Department, New Jercy Institute of Technology, 2001.

[6] W. Fan, M. D. Gordon, P.Pathak, W. Xi, E. A. Fox, "Ranking Function Optimization for Efficient Web Search By Genetic Programming : An Empirical Study", Dept. of Computer Science of Virginia Tech, Michigan, Florida Universities, 2003.

[7] K. Kozhuharov, G. Georgiev, M. Nagaa, M. Omiya, "Implementation of a XML-Web Service for Distributed Genetic Algorithms", Hokkaido University, 2003.

[8] Y. Xie, D. Mundlura, V. V. Raghavan, "Incorporating Agent Based Neural Network Model for Adaptive Meta-Search", The Center for Advanced Computer Studies, University of Louisiana at Lafayette, USA, 2004.

[9] www.Searchenginewatch.com, Visited at 2006.

[10] E. Bonabeau, M. Dorigo, G. theraulaz, "Inspiration for Optimization from Social insect Behavior", Nature, 406, 39-42, 2000.

[11] S. Lawrence, L. Giles, "Accessibility of Information on the Web", Nature 400, 107-109, Nec Research Institute, 1999.

[12] W. Meng, C. Yu, and K. Liu, "Building Efficient and Effective Meta-Search Engines", ACM Computing Surveys, vol. 34, no. 1, March 2002, pp(48-89).

[13] J. Barker, "Meta-search Engines", Teaching Library Internet Workshops University of California, Berkely, April, 2000.

[14] S. Raghavan, HG. Molina, "Crawling the Hidden Web", VKDB Conference, pp 129-138, Italy, 2001.

[15] Y. Fan, and S. Gauch, "Adaptive Agents for Information Gathering from Multiple, Distributed Information Sources", In Proceedings of 1999 AAAI Symposium on Intelligent Agents in Cyberspace, Stanford University (March 1999), pp40-46.

[16] S. Zhu, X. Deng, K. Chen, and W. Zheng, "Using Online Relevance Feedback to Build Effective Personalized Meta-Search Engine", Proceedings of Second International Conference in Web Information Systems Engineering, 2001.

[17] J. Choi, M. Kim, and V. V. Raghavan, "Adaptive Feedback Methods in an Extended Boolean Model", In proceedings of ACM SIGIR Workshop on Mathematical/Formal Methods in Information, New Orleans, LA, Sept. 2001.

[18] A. Araus, et. al., "Searching the Web", ACM Transactions on Internet Technology, Vol. 1, August 2001, pp:2-43.

[19] J. W. Green, "Hyper Dog : Up to Date Web Monitoring through Meta Computers", MS Report, Baltimore, Maryland, October 2000.

[20] S. Lawrence, L. Giles , "Searching the World Wide Web", Science, VOL. 280, No. 5360, April 1998, pp:98-100.

[21] W. May, G. Lausen, "Information Extraction from the Web", TR. No. 136, Institute for Informatic, Albert-Ludwings University, Germany, March 2000.

[22] M. Kobayashi, K. Takeda, "Information Retrieval on the Web", ACM Computing Surveys, Vol. 32, Issue 2, June 2000, pp:144-173.

[23] S. Brin, L. Page, "The Anatomy of large Scale HyperTextual Web Search Engine", URL:www7.scu.edu.au/programme/fullpapers/1921/com1021.htm Visited at 2006.

[24] Talim et al, "Controlling the Robots of Web Search Engines", Proceedings of ACM SIGMETRICS 2001, pp:236-244.

[25] V. Gupta, R. Campbell, "Internet Search Engine Freshness by Web Server Help", TR-UIUCDCS-R-2000-2153, Digital Computer Library.

[26] J. Barker, "Meta-Search Engines", Teaching Library Internet Workshops University of California, Berkeley, April 2000.

[27] B. Grossan, "Search Engines : What they are, How they work, and Practical Suggestions for Getting the most out of them", February, 1997, http://www.webreference.com/content/search.

[28] M. Henzinger, "Web Information Retrieval", 16th International Conference on Data Engineering, IEEE Computer Society, San Diego, CA, USA, February 29-march 3, 2000.

[29] A. Kingoff, "Comparing Internet Search Engines", Computer(30:4), April 1997, pp.117-118.

[30] I. Winship, "Web Search Service Features", February 2001, http://www.unn.ac.uk/central/isd/features.htm.

[31] G. S. Goldsmidt, "Distributed Management by Delegation", Ph.D-Thesis, Columbia University, 1996.
[32] D. B. Lang, and M. Oshima, "Programming and Developing Java Mobile Agents with Aglets", Addison Wesley, Menlo Park, CA, August 2002.
[33] D. B. Lang and M. Oshima, G. Karjoth and K. Kpsaka, "Aglets : Programming Mobile Agents in Java", Proceedings of the International Conference, Tsakaba, Tapan (March 1997), Lecture Notes in Computer Science 1274, Springer-verlag, Berlin, Germany, pp.253-266.
[34] Y. Wany, D. J. Dewitt, "Computing PageRank in a Distributed Internet Search System", Computer Sciences Department, University of Wisconsin, Madison, USA, Proceedings of the 30th VLDB Conference, Toronto, Canada, 2004.
[35] S. Zabala G. Loerics, y. Bello V. Dias, "CALVIN : A Personalized Web-Search Agent-Based on Monitoring User Actions", Artificial Intelligence Group, Caraca, Venezoela, 2000.
[36] Forrelly, Glen, "Search Engines : Evolutions and Revolution", July 1999 http://Webhome.idirect.com~glenjenn/search/history.htm.
[37] Y. Fan, "An Adaptive Multi-Agent Architecture for the ProFusion* Meta Search System", Department of Electronic Engineering and Computer Science University of Kansas, Proc. Of WebNet'97, Nov.,1997.
[38] D. B. Lange, "Java Aglet Application Programming Interface (J-AAPI)", IBM Tokyo Research Laboratory February 19, 1997.
[39] Z. Z. Nick, P. Themis, "Web Search Using a Genetic Algorithm", University of Piraeus, IEEE Internet Computing, 2001.
[40] M. Sanan, "Smart Search Engines", University of Caen, France, 2005.
[41] T. Finin and L. Ding, "Search Engines for Semantic Web Knowledge", University of Meryland, USA, Proceedings of XTech,2006.
[42] M. Biddulph, "Crawling the Semantic Web", Proceedings of XML Europe, 2004.
[43] D. F. Barrero, "SEAECHY: A Aetasearch Engine for Heterogeneous Sources in Distributed Environments", University of Alcala, 2004.
[44] D. F. Barrero, D. R. Lopez and O. Garcia, "Distributed Metainformation Searching: An Approach to Information Retrieval in the age of the Semantic Web", In VII TERENA Networking Conference,2004.
[45] "Web Search Environment", http://wse.search.ac.uk/demo.htm, Visited in 2006.
[46] L. C. Kingsland, M. F. Prettyman, S. E. Shooshan, "The NLM Gateway: A Metasearch Engine for Disparate Resources", MEDINFO 2004, Amsterdam: IOS Press.
[47] R. Beaza-Yates, "International Retrieval in the Web: Beyond Current Search Engines", Center for Web Research, Department of Computer Science, University of Chile, Blanco Encalada, 2120 Santiago, Chile, 2003.
[48] H. Tirri, "Search in Vain: Challenges for Interest Search", IEEE Computer, 2003.
[49] J. Greg, "Meta Search Engines are Bak", www.searchenginewath.com, Visited in 2006.
[50] B. Joe, "What are Meta Search Engines? How do they work? Finding Information on the Internet", http://www.lib.berkely.edu/TechingLib/Guides/Internet/Metasearch.html, Visited in 2006.
[51] "Search Engine Shawdown", http://www.searchengineshowdown.com/newsarchive/000765.shtml, Visited in 2006.