Paper

Effectiveness of Mobile Agent for Query Retrieval

Member	Ali Selamat	(Osaka Prefecture University)
Member	Sigeru Omatu	(Osaka Prefecture University)
Member	Hidekazu Yanagimoto	(Osaka Prefecture University)
Member	Toru Fujinaka	(Osaka Prefecture University)
Member	Michifumi Yoshioka	(Osaka Prefecture University)

Information searching and retrieving are two important aspects that most of the Internet users do when they are using the Internet. Unfortunately, many available tools that are being used to search a particular information from the Internet such as a search engine could not give the satisfying results. This is due to the problem such as broken and duplicated links. Also the ranking algorithms that are applied by the search engines are different from one to another. This paper discusses the limitations of current software tools to search in the Internet and proposes a new approach by employing a mobile agent to retrieve the query results in the local search servers. The Mobile Agent Search System (MaSS) has been developed to reduce the time for searching and retrieving information from the Internet. Furthermore, the ranking of query results are evaluated using the Number of Relevant Ordering Score ($NROS$) method. The main advantage of the MaSS is to reduce the time for searching and retrieving information as it is done in the off-line case compared to the on-line case.

Keywords: mobile agents, information retrieval, web agent, World Wide Web, data mining.

1. Introduction

The evolution of information age has enabled users to search and retrieve many types of information from the Internet. Special application softwares have been introduced to overcome the problem in searching information from the Internet which is the search engine. The main purpose of the search engine is to make an indexing process on a million of web pages that exist in the entire World Wide Web (WWW) [1]~[3]. The list and content of URLs are stored in a local search engine database or indexing file. This is the file that will be referred by the search engine when a user sends a request. There are many search engines that exist today such as Yahoo, AltaVista, and HotBot (abbreviated as search engines). As the search engines are useful for a user to search the desired information from the Internet, the following problems related to the query retrieval results have remained,

- The contents of the web pages that are being indexed by the search engine have been already backdated. A user needs fresh information from the search results to be classified as a useful information.
- The query results are not relevant to a user interests.
- To acquire multiple query results, a user needs to search each of the search engines. As a result, the user has to spend more time on searching and navigating the Internet.
- The broken links that exist on some of the results from the search engine have made a user frustrated with the results.
- Each of the search engines has a different ranking algorithm. Therefore, important results that the user needs could be ranked differently by each search engine.
- The duplicated links that exist in the search engine results made a user uncomfortable to browse the same page for many times.

A user still has not been satisfied with the search results that have been given by a search engine due to the above reasons [4]. To retrieve the query results and preprocess them locally, it will take some time before a user can actually receive the query results. Also there is a need to prefetch the query results and filter them locally before they are presented to a user. Therefore, this paper solves the above problems using a mobile agent technology for query document retrievals from the Internet. Furthermore, we have developed the MaSS to support the query retrieval process. The ranking of query results is evaluated using the $NROS$ method.

In Chapter 2 the description of mobile agent and query retrieval are explained. In Chapter 3 the architecture and evaluation of the MaSS are described. In Chapter 4 the ranking of query results and the evaluation using the $NROS$ method are discussed. In Chapter 5 the discussion and the conclusion are stated.

2. Mobile Agent and Query Retrieval

A mobile agent [5] [6] is an autonomous program that can move from one machine to another in a heterogeneous network under its own control. It can suspend its execution at any point, transport itself to a new machine from the point that its left out. On each machine, it interacts with the service agents and other resources

to accomplish its task. Then it returns to its home site with final results when the task has finished. A few advantages highlighted using mobile agents are to reduce network loads, overcome network latency, encapsulate protocols, and to execute a program asynchronously and autonomously. Further discussion on the advantages of mobile agents are described by Lange et al. [7]

2.1 Query retrieval using mobile agent We have applied the mobile agent technology for prefetching the results from the search engines. The term prefetching means that we have used our mobile agents to retrieve the URL datasets that exist in the search engines databases and store them in the local database servers. The process has been done during night-time by employing the activation and deactivation functions that exist in our mobile agent system. Activation is a process of activating the mobile agent to start retrieving the search engine databases based on specific queries. In our experiment we have set the number of queries to 30 as in Table 1. Also we have set the activation time to 1-hour where in each hour, the mobile agent will retrieve the query results from the designated search engines. If the search query result is new, then it will be added to the local database, otherwise, it will be discarded. A deactivation is a process to stop the mobile agent from collecting the new search results, which will be determined by the user. Further description on mobile agent approach to the system will be described in the next chapter.

3. The Architecture and Evaluation of the Mobile Agent Search System (MaSS)

We have developed the Mobile Agent Search System (MaSS) to support retrieval query results. The general architecture of the MaSS is shown in Fig. 1. The MaSS architecture consists of a MaSS client, a MaSS server, and a collection of the local prefetch servers. Figure 2 represents the detail architecture of off-line retrieval using the MaSS. The on-line retrieval using the MaSS is shown in Fig. 3. Further description on each component of the MaSS is described as follows:

3.1 The MaSS client The MaSS client comprises of the MaSS client agent and a simple user interface. A user interface is used as a medium for a user to interact with the MaSS.

3.1.1 The MaSS client agent A user will enter a search keyword on the query form in the HTML browser. The MaSS client agent will send a request to the MaSS server agent in order to get the query search results. The MaSS client agent is represented as W in Fig. 2.

3.2 The MaSS server The MaSS server comprises of the MaSS server agent and the MaSS search agent. Both of them are represented as X and Y in Fig. 2. Further descriptions on each of them are as follows:

3.2.1 The MaSS server agent The MaSS server agent is a stationery agent. Upon receiving request from the MaSS client agent, the user's id will be checked at this stage. Then the MaSS server agent will delegate the search tasks to the MaSS search agent. After receiving the search results from the MaSS search agent, the MaSS server agent will rank them using the *NROS* method as described in Chapter 4. The ranked search results will be stored into the MaSS server database before returning them to the MaSS client agent to be presented to the user.

3.2.2 The MaSS search agent When receiving a query request from the MaSS server agent, the MaSS search agent will start to mobile to a collection of local prefetch servers as shown in Fig. 2. Then it will communicate with the local agents (Z1, Z2, and Z3) at each host B, C, and D. At host B, the MaSS search agent will ask the local agent (Z1) about the query that is requested by the MaSS server agent. The query results will be given to the MaSS search agent by a local agent (Z1). The same process will be repeated at hosts C and D with different local agents (Z2 and Z3). Once the tasks have been completed, the MaSS search agent will return home and pass the search results to the MaSS server agent.

3.3 A collection of local prefetch servers The operational architecture of the MaSS in retrieving the search results from the collection of local prefetch servers in an off-line mode is shown in Fig. 2. The local prefetch servers consist of a local Yahoo database server (B), a local AltaVista database server (C), and a local HotBot database server (D). The idea of local prefetch server is to store the query results locally instead of searching them from the WWW. They will reduce the time of searching and retrieving the query results by the MaSS search agent. The local prefetch servers will send their local agents (Z1, Z2, and Z3) to roam the WWW and collect the search results from designated search engines. In our case, we have selected 30 queries from each search engine as in Table 1. Each local agent (Z1, Z2, and Z3) will fetch the 200 results on each of the query. These local agents will store the query results in a local database at hosts B, C, and D. The total of query results are 200 query results $\times$ 30 types of query (see Table 1) = 6,000 pages. The process of removing the broken and duplicated links, stemming, and stopping will take place before storing them into a local database. This is done in a night-time using activation and deactivation function that exist in the local agents (Z1, Z2, and Z3).

3.4 The off-line retrieval process The off-line retrieval process is presented in Fig. 2. When accepting a request from the MaSS client agent (W), the MaSS server agent (X) will pass the query item to the MaSS search agent (Y). The MaSS search agent will start its routes from A $\rightarrow$ B to collect the query results at the local prefetch server B. At B, the MaSS search agent will ask the local agent (Z1) for the search results. Once completed, it continues its search to the local prefetch server C. At C, the same process is repeated with local agent Z2. Then the MaSS search agent will continue its search at server D. The MaSS search agent will return to location A and pass the results to the MaSS server agent. The MaSS server agent will rank the search

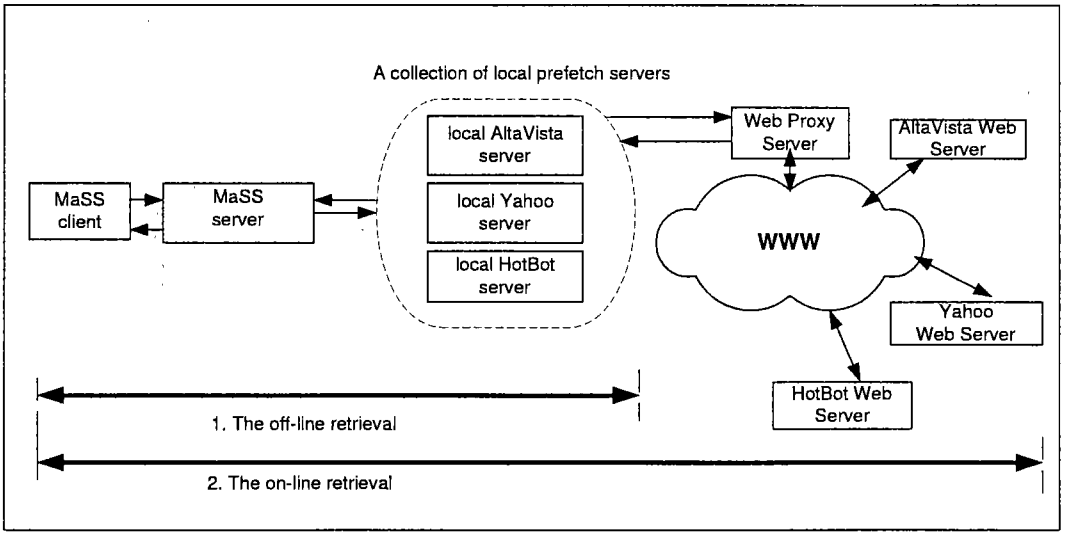


Fig. 1. The Mobile Agent Search System (MaSS) architecture.

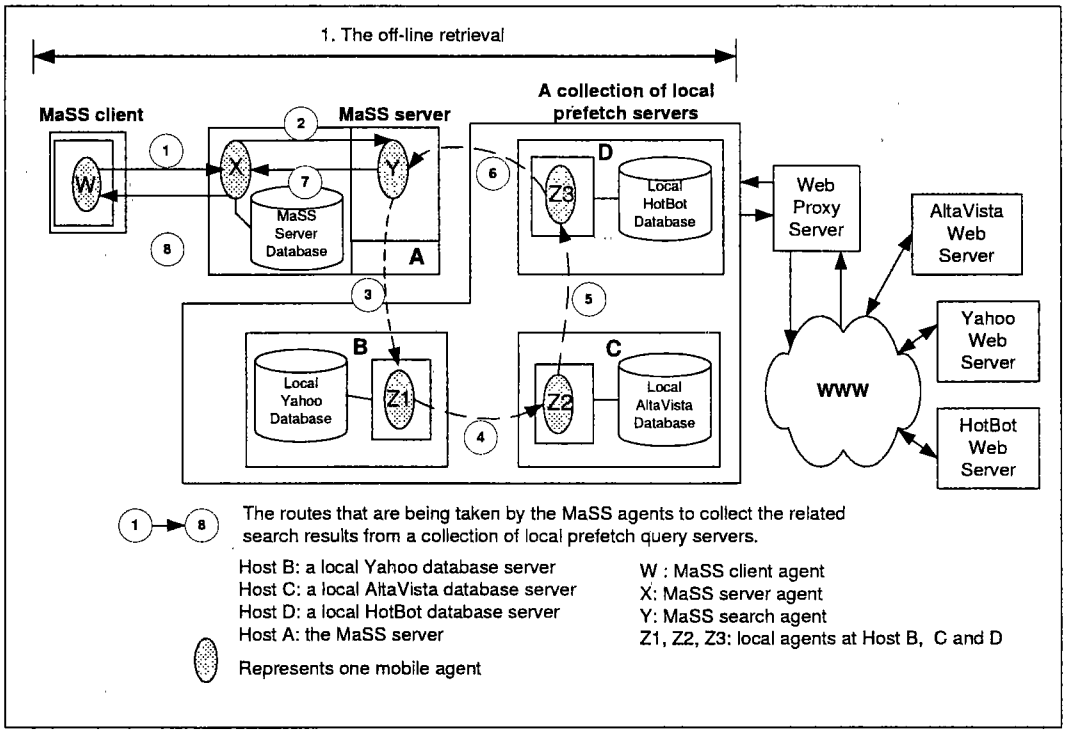


Fig. 2. The implementation of mobile agent for the off-line fetching in the MaSS.

query results and store them in its local database. Then the search results will be sent to the MaSS client agent to be presented to the user.

3.5 The on-line retrieval process The on-line retrieval process is presented in Fig. 3. When accepting a request from the MaSS client agent (W), the MaSS server agent (X) will pass the query item to the MaSS search agent (Y). Instead of looking for the prefetch servers, the MaSS search agent will search in the WWW and retrieve the search results from the search engines. The search results will be stored into the MaSS server database. Here, the process of removing duplicated and broken links, stemming, stopping, and ranking the search results will take place.

3.6 The MaSS evaluation We will evaluate the query retrieval results using the *NROS* method that will be described in the following Chapter 4. The MaSS is evaluated using the on-line and off-line retrieval strategies that are described as below.

3.6.1 Web fetching We have used a WebL script programming language from Compaq [8] to develop a web-fetching program. The script program will download the HTML files from the WWW. The results will be sent to the user using an Aglet mobile agent.

3.6.2 The on-line retrieval evaluation As the evaluation of the off-line retrieval technique, the MaSS

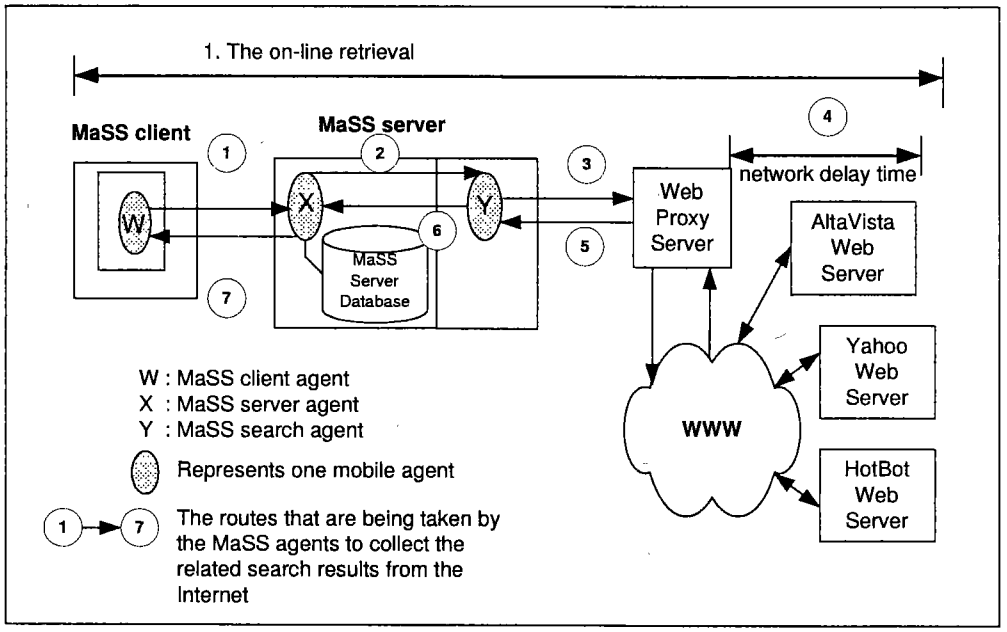


Fig. 3. The implementation of mobile agent for the on-line fetching in the MaSS.

Table 1. The comparison of the time taken by the MaSS using on-line and off-line mode with 30 query cases.

Query	Doc. Size(kb)	CPU time taken for the on-line retrieval process (ms)	Total of downloading time using the on-line retrieval process(ms)	Total of downloading time using the off-line retrieval process(ms)
1. Computer engineering	6,078	227,303	229,287	1,984
2. Mercedes Benz,Volvo and Fiat	5,535	380,003	382,924	2,921
3. Visiting Australia and Japan.	4,640	677,661	678,242	581
4. The Japan and European Garden	4,380	456,793	458,045	1,252
5. Cancer or body disease	5,819	341,996	343,814	1,818
6. Flower arrangements, origami, Japanese Culture	6,445	744,275	745,740	1,465
7. Food and Beverages in Osaka	3,652	173,387	173,893	506
8. Biocomputing and evolutions	3,128	165,294	166,115	821
9. Visiting German and Europe	2,915	176,812	177,833	1,021
10. Cooking Books	3,251	211,725	213,188	1,463
11. Human Computer Interaction or HCI	2,603	173,757	174,263	506
12. Java	2,890	364,751	366,624	1,873
13. Aglets	2,675	255,463	2,656,184	721
14. Data Mining and Farming	2,322	720,81	73,242	1,161
15. Distributed Programming	3,669	169,571	170,497	926
16. Cancer Preventions	3,803	133,826	134,090	264
17. Gordon Blair	4,438	539,980	542,782	2,802
18. The Times Magazine	2,568	112,734	116,609	3,875
19. The World Newspaper	2,644	199,950	203,134	3,184
20. George Bush, US President	3,275	174,703	179,700	4,997
21. Conferences in Neural Networks	3,527	259,726	264,365	4,639
22. World Economic Forums	2,589	202,143	205,373	3,230
23. The world economic recession and its impact	1,706	728,55	75,975	3,120
24. Car manufacturers	2,725	165,320	169,223	3,903
25. Chemical engineering products	2,644	199,950	203,134	3,184
26. Japan Industrial Products	3,275	174,703	179,700	4,997
27. Supply and Chain Management	3,527	259,726	264,365	4,639
28. Education in United States	2,589	202,143	205,373	3,230
29. Living in France	1,706	728,55	75,975	3,120
30. Algorithms and Data Structure	2,725	165,320	169,223	3,903

server agent will send users request to the MaSS search agent. It is referring to the total time taken from (1)→(7) as shown in Fig. 3. The MaSS search agent will retrieve the query results from the search engines in an on-line mode through the WWW. The removal process of duplicated or broken links and ranking the query results are done at the MaSS server. The CPU processing time is the time taken by the CPU to process the query results. The processes involved in the on-line retrieval of web pages have been described in Section

3.5. As these processes are done in a local MaSS server, the time taken to process the query and send the results to a user is significantly long compared to the off-line retrieval (see Table 1). Also the network delay as in Fig. 3, contributes the time taken to finish the retrieval process.

3.6.3 The off-line retrieval evaluation The off-line retrieval process is evaluated by the total time that the users have to send a query to the MaSS server agent and the turn-around time to finish the retrieval process. The turnaround time is the time from the user agent to the MaSS server agent, and from the MaSS server agent to the MaSS search agent, the waiting time of the MaSS search agent, and the return-time to the user. It is referring to the total time taken from (1) → (8) as shown in Fig. 2. The waiting time is the time taken to complete the query process. The downloading time is low compared to the on-line retrieval process as described in previous section. This is due to the reason that the processes of creating the term frequency matrix, stemming, and stopping have been done during the prefetching process by the collection of local prefetch servers.

Table 2. The $length_j$ is calculated by the total values of TF_i in Doc_j.

Doc_j	TF_1	TF_2	...	TF_N	$length_j$
Doc_1	2	4	...	5	300
Doc_2	2	3	...	2	340
⋮	⋮	⋮	⋮	⋮	⋮
Doc_n	2	3	...	5	326

4. The Ranking and Evaluation of Query Results Using the NROS Method

The ranker module calculates the *Inverse Document Frequency (IDF)* [9] and similarity indices for the entire set of URLs candidate and sorts them in order. But this is the basic calculation for measuring the query search results, where the broken and duplicated links parameters are not considered. The *IDF* measures the goodness of each term frequencies, therefore, acts as a document discriminator. In other words, the measure highlights the document in which the terms occur. The *IDF* is given by the following equation given by Salton [10].

$$IDF_i = \frac{\log_2(Max\ \ n_i) - 1}{n_i} \quad \cdots\cdots\cdots\cdots\cdots (1)$$

where n_i is the total number of occurrences of term i in the collection and $Max\ n_i$ is the maximum frequency of any term in the collection. Term frequencies within documents are obtained by the *IDF* measure. The similarity measure [10], which is a form of document frequency weight, is used to rank the documents in the MaSS. The similarity measure is given by

$$Similarity_j = \frac{\sum_{k=1}^{Q}(\log_2(TF_{ij} + 1)) \times IDF_i}{\log_2(length_j)} \quad \cdots (2)$$

where TF_{ij} is the term frequency of word i in document Doc_j where $i = 1, \ldots, N$ and $j = 1, \ldots, n$. The $length_j$ is the total number of unique terms in Doc_j after the stopping and stemming processes. The Q in eq. (2) is the total number of queries. Once the similarity scores for each document have been obtained, the list is sorted in descending order. The scores will be included in the calculation of the ranking method as described in Section 4.1. We illustrate the example of $length_j$ as in Table 2. The $length_j$ is given by

$$length_j = \sum_{i=1}^{N} TF_{ij}. \quad \cdots\cdots\cdots\cdots\cdots\cdots (3)$$

4.1 The NROS method This section has described the *NROS* method that has been applied to the MaSS query ranker. We have included the broken and duplicated links parameters in this method. Table 3 shows the query results on *'computer engineering'*, before we apply our *NROS* method. n is the total number of Doc_j that has been retrieved by the MaSS search agent. The $Similarity_j$ is calculated based on eq. (2). We have used a value of 1 if the link is duplicated and 0 otherwise. Also a value of 1 has been used if the link is broken and 0 otherwise. The *Erroneous* links are given by

$$Erroneous = \frac{(D + Error)}{R} \times (1/100) \quad \cdots\cdots (4)$$

where *Erroneous* represents the number of URLs that have broken and duplicated links. D represents the total number of duplicated documents that exist in the results set, and R is the total of similarity score. *Error* represents the total number of broken links that exist in the results set. The value of 1/100 has been used to normalize the value of *Erroneous*. Referring to Table 3, the value of $n = 16$, $R = 13.58509$, $Error = 2$, and $D = 7$. The value of *Errorneous* is 0.00662.

Then we have defined the Relevant Ordering Score (*ROS*), as the order of documents with respect to the total number of query retrieval results and the *Erroneous*. *ROS* is given by

$$ROS = \frac{\sum_{j=1}^{n} Similarity_j}{n} - Erroneous. \quad \cdots\cdots (5)$$

As for an example, the value of *ROS* in Table 3 is 0.84244. For a set of query results as in Table 3, the *ROS* is a constant value used for the *NROS*. The *NROS* is calculated based on the value of *ROS* and the similarity measures as in eq. (2). It is a value used for ranking the query results after the process of identifying the broken and duplicated links. If an error due to broken links or duplicated links occurs on the query results, the penalty of *ROS* is given to the Doc_j. The *NROS* is given by

$$NROS_j = Similarity_j - ROS \quad \cdots\cdots\cdots\cdots (6)$$

Table 3. The query results on *'computer engineering'* by the MaSS search agent before using the *NROS* method.

Doc_j $(j = 1, \ldots, n)$	Similarity Score $(Similarity_j)$	URL links	Duplicated links (D_j)	Broken links $(Error_j)$
1	1.56726	http://www.ece.ufl.edu/	0	0
2	1.24102	http://www.ece.utexas.edu/	0	0
3	1.19769	http://www.computer.org/	0	0
4	1.19438	http://www.ece.cmu.edu/	0	0
5	1.11876	http://www.cs.washington.edu/	0	0
6	1.03826	http://www.sdsc.edu/	1	0
7	0.74731	http://www.sdsc.edu/	1	0
8	0.72849	http://www.sdsc.edu/	1	0
9	0.71265	http://www.sdsc.edu/	1	0
10	0.68851	http://www.ece.gatech.edu/	0	0
11	0.66956	http://www.ece.uiuc.edu/	0	0
12	0.65781	http://www.ee.ualberta1.ca/	0	1
13	0.63754	http://www.ece.uwaterloo.ca/	1	0
14	0.55918	http://www.ece1.ncsu.edu/	0	1
15	0.50554	http://www.ece.uwaterloo.ca/	1	0
16	0.32111	http://www.ece.ufl.edu/	1	0

$n = 16$; $R = \sum_{j=1}^{n} Similarity_j = 13.58509$; $D = \sum_{j=1}^{n} D_j = 7$; $Error = \sum_{j=1}^{n} Error_j = 2$.

where $NROS_j$ is the value of $NROS$ for query retrieval results of Doc_j. The $Similarity_j$ is referring to the similarity value of Doc_j as in eq. (2). The ROS is given from eq. (5). If there is no duplicated or broken links that exist in the results set, the $NROS_j$ value will be the same as $Similarity_j$. We have applied the $NROS$ method to the query results that have been retrieved. The higher score of the $NROS$ indicates that the document is more relevant to the users request. The final results that consist of the ranking value of each link are presented to the user as in Table 4.

Table 4. The query results on *'computer engineering'* by the MaSS search agent after using the *NROS* method.

$NROS_j$	Doc_j $(j = 1, \ldots, n)$	URL links (D_j)
1.56726	1	http://www.ece.ufl.edu/
1.24102	2	http://www.ece.utexas.edu/
1.19769	3	http://www.computer.org/
1.19438	4	http://www.ece.cmu.edu/
1.11876	5	http://www.cs.washington.edu/
0.68851	10	http://www.ece.gatech.edu/
0.66956	11	http://www.ece.uiuc.edu/
0.19508	6	http://www.sdsc.edu/
-0.09587	7	http://www.sdsc.edu/
-0.11469	8	http://www.sdsc.edu/
-0.13053	9	http://www.sdsc.edu/
-0.18537	12	http://www.ee.ualberta1.ca/
-0.20564	13	http://www.ece.uwaterloo.ca/
-0.28399	14	http://www.ece1.ncsu.edu/
-0.33764	15	http://www.ece.uwaterloo.ca/
-0.52207	16	http://www.ece.ufl.edu/

5. Discussions and Conclusions

5.1 Discussions The effectiveness of the *NROS* is evaluated using standard information retrieval measures that are *RECALL* and *PRECISION* [10]. *RECALL* is the ratio of the number of relevant records retrieved to the total number of relevant records in the database. *PRECISION* is the ratio of the number of relevant records retrieved to the total number of irrelevant and relevant records retrieved. They are given by

$$RECALL = A/(A + B) \quad \cdots\cdots\cdots\cdots\cdots\cdots (7)$$

$$PRECISION = A/(A + C) \quad \cdots\cdots\cdots\cdots\cdots\cdots (8)$$

where A is the number of relevant record retrieved, B is the number of relevant record not retrieved, and C is the number of irrelevant record retrieved. In our experiments we have found that the average of *PRECISION* is 82% and the average of *RECALL* is 66% (see Table 5). A high *PRECISION* indicates that from all of the documents returned by our query, a large proportion of the documents is relevant to the search. A high *RECALL* indicates that from all of the documents in the archive that are relevant to the query, a large number of these documents are returned.

As for the mobile agent, it provides significant time saving using an off-line retrieval process for sending results from the MaSS server to the MaSS client compared with the on-line retrieval process. The detail of query types is included in Table 1. For example, if a user makes a query on *'computer engineering'*, using the on-line retrieval process, the total time taken for a user to wait until the results are received is 229,287ms. Comparatively, by using the off-line retrieval process, the time that the user has to wait until the query results are received is 1,984ms. Also for the query title *'Visiting Australia and Japan'*, the time taken using the MaSS in the off-line retrieval process is 581ms. In comparison, the time taken using the on-line retrieval process is 678,242ms. Further examples of the time taken by user to be on-line and off-line with different queries are shown in Table 1.

5.2 Conclusions The MaSS has been designed to reduce the information overload problem as well as the Internet Connection Time. A collection of local prefetch servers enables us to store and preprocess locally the query results before presenting them to the user. The effectiveness of the MaSS is evaluated using

the off-line and on-line retrieval strategies. We have found that the time spend for the off-line retrieval is less compared with the on-line retrieval. The effectiveness of the *NROS* method is evaluated using standard information retrieval measures that are *RECALL* and *PRECISION*. The duplicated and broken links removal, and ranking the query results processes have increased the *PRECISION* and *RECALL* measures of the search results.

Table 5. The evaluation of *RECALL* and *PRECISION* of the query results that being retrieved by our method

Query Number	*RECALL*(%)	*PRECISION*(%)
Q1	80	88
Q2	83	88
Q3	75	92
Q4	73	85
Q5	77	87
Q6	80	90
Q7	77	76
Q8	73	74
Q9	64	88
Q10	66	80
Q11	60	75
Q12	60	75
Q13	67	73
Q14	60	67
Q15	64	78
Q16	64	78
Q17	60	86
Q18	70	88
Q19	64	88
Q20	60	85
Q21	70	90
Q22	70	90
Q23	60	86
Q24	60	87
Q25	60	88
Q26	70	88
Q27	70	72
Q28	64	70
Q29	50	80
Q30	40	75
Average	66	82

(Manuscript received February 26, 2001, revised February 15, 2002)

References

(1) O. E. Zamir, "Clustering Web Documents: A Phrase-Method for grouping Search engine Results," PhD Thesis, University of Washington, 1999.

(2) MetaCrawler: http://www.metacrawler.com, 1999.

(3) M. E. Muller, "An Intelligent Multi-Agent Architecture for Information Retrieval from the Internet," Technical Report, Institute for Semantic Information Processing, University of Osnabruck, Germany, 1999.

(4) C. Wai Ho and A. Goh, "Jamaica: A World Wide Web Profiler," Internet Research: Electronic Networking Applications and Policy, Vol 9: Number 2, pp. 129-139, 1999.

(5) TACOMA: http://www.cs.uit.no, 1997.

(6) P. Maes, Y. Lashkari, and M. Metral, "Collaborative Interface Agents,", In: Readings in Agents, edited by Michael N. Huhns & Munidar P. Singh, Morgan Kaufmann Publishers, Inc., 1997.

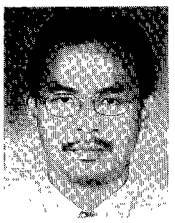
(7) D. Lange and M. Oshima, "Programming and Developing Java Mobile Agents With Aglet Mobile Agent," Addison Wesley, 1998.

(8) Compaq Web Language: http://www.research.compaq.com/SRC/WebL/, 1997.

(9) R. R. Korfhage, "Information Storage and Retrieval," John Wiley and Sons, Inc, USA, 1997.

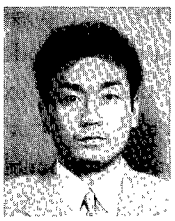
(10) Salton & McGill, "Introduction to Modern Information Retrieval," New York, McGraw-Hill, USA, 1983.

Ali Selamat (Member) He received B. Sc. in IT from Teesside University, U.K. and M. Sc. in Distributed Interactive Systems from Lancaster University, U.K. in 1997 and 1998 respectively. He was with University of Technology Malaysia as an assistant professor from 1999-2000.Currently, he is pursuing his Ph. D. at Graduate School of Engineering, Osaka Prefecture University, Japan. His research interests include mobile agent and multi-agent technology, web farming, data mining and Genetic Algorithm.

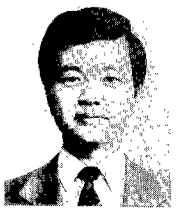


Sigeru Omatu (Member) He was born on December 16, 1946. He received the B. E. degree in Electrical Engineering from the Ehime University in 1969 and the M. E. and Ph. D. degrees in Electronics Engineering from Osaka Prefecture University in 1971 and 1974, respectively. He was with the Tokushima University as a research associate since 1974, a lecturer since 1975, an associate professor since 1980, and a professor since 1988. Then he joined the Osaka Prefecture University in 1995, where he is currently a professor in the Graduate School of Engineering.

Hidekazu Yanagimoto (Member) He was born on March 14, 1972. He received the B. E. and M. E. degrees from Osaka Prefecture University in 1994 and 1996 respectively. He joined Human Media Research Laboratories, NEC, Japan in 1996. Since 2000 he has been a research associate in the Graduate School of Engineering, Osaka Prefecture University, Japan. His research interests include Information Retrieval and Genetic Algorithm.



Toru Fujinaka (Member) He received the B. E., M. E., and Ph. D. degrees, all in Electrical Engineering from Kyoto University in 1979, 1981, and 1990, respectively. He was with Kyoto University as a research associate from 1986 to 1990, then he joined Osaka Prefecture University, where he is currently an assistant professor in the Graduate School of Engineering. His research interests include theory and application of optimal and intelligent control.



Michifumi Yoshioka (Member) Michifumi Yoshioka was born in Osaka, Japan, on Dec., 10, 1968. He received the B. E., the M. E, and Ph. D. degrees in Geo System Engineering from University of Tokyo, Japan, in 1991, 1993, and 1996, respectively. Since 1996, he has been a Research Associate at Osaka Prefecture University. His current interests center on image processing methods using neural networks.