VOT 75077

NETWORK ROUTING BY MOBILE AGENTS FOR QUERY RETRIEVAL USING GENETIC ALGORITHMS

(PENJELAJAHAN RANGKAIAN OLEH AGEN MOBIL BAGI CAPAIAN PERTANYAAN MENGGUNAKAN ALGORITMA GENETIK)

DR. ALI SELAMAT
MD. HAFIZ BIN SELAMAT
MD. SAH BIN HAJI SALAM
SARINA BT. SULAIMAN
ARYATI BT. BAKRI

FAKULTI SAINS KOMPUTER & SISTEM MAKLUMAT
UNIVERSITI TEKNOLOGI MALAYSIA

2007

ACKNOWLEDGEMENT

We would like to extend our appreciation to Research Management Center, Universiti Teknologi Malaysia for funding this project (Vote No. 75077).

ABSTRACT

Mobile agents often have a task to collect data from several predefined sites. This should be done in an efficient way by minimizing the elapsed time. Usually these agents only know the list of sites but not the distances between them. This research proposes a method to minimize a network routing time taken by the mobile agents using genetic algorithm (GA) to collect query datasets from different hosts from the World Wide Web. The mobile agents repeat traveling over short routes and avoid longer ones. The GA will be used to select the best routes that will be used by mobile agents to retrieve query datasets in a very short time. Specifically, the performance of mobile agent in obtaining a query result from the remote hosts using extended hierarchical query retrieval (EHQR) approach is proposed. It is based on a hierarchical and a parallel dispatching of mobile agents to the remote servers in order to retrieve the query results. the new dispatching algorithms for mobile agents to be dispatched to various sited have been developed such as an itinerary query retrieval approach (IQR), a parallel query retrieval approach (PQR), a serial query retrieval approach (SQR), a shuttle query retrieval approach (SHQR), a hierarchical query retrieval approach (HQR) and the EHQR approach Experimental results show that the proposed approach reduces the number of mobile agents and also improve the total time taken to retrieve the query results compared with other approaches.

ABSTRAK

Pada kebiasaannya, agen-agen mobil ditugaskan untuk mengumpul data daripada beberapa laman web. Ini perlu secara efisien bagi mengurangkan masa yang diambil bagi proses tersebut. Biasanya agen-agen ini hanya mengetahui senarai laman-laman web yang dilawati tetapi tidak mengambil kira jarak di antara mereka. Dengan itu, penyelidikan ini mencadangkan satu kaedah bagi meminimumkan masa sesebuah saluran rangkaian diambil oleh agen-agen mobil dengan menggunakan algoritma genetik (GA) untuk mengumpul set-set data pertanyaan daripada pelayan web yang berbeza daripada dunia jaringan luas. Di mana, agen-agen mobil akan mengulangi semula proses melawati laman-laman web tersebut dengan menggunakan laluan-laluan pendek dan mengelakkan agen dari melalui laluan yang lebih panjang. Oleh itu, GA akan digunakan bagi memilih laluan-laluan terbaik untuk digunakan oleh agen-agen mobil dalam mendapatkan semula set-set data pertanyaan dalam masa yang singkat. Secara khususnya, keupayaan agen mobil dalam mendapatkan hasil pertanyaan daripada pelayan menggunakan pendekatan capaian pertanyaan yang berhierarki dan terlunjur (EHQR) adalah dicadangkan. Ini dilakukan berdasarkan penghantaran agen-agen mobil secara hierarki dan selari ke pelayan-pelayan bertujuan untuk mendapatkan semula hasil capaian pertanyaan. Terdapat pelbagai algoritma baru telah dibangunkan bagi proses penghantaran agen-agen mobil ke pelbagai pelayan, contohnya pendekatan jadual capaian pertanyaan (IQR), pendekatan capain pertanyaan secara selari (PQR), pendekatan capaian pertanyaan secara bersiri (SQR), pendekatan capaian pertanyaan secara ulang alik (SHQR), pendekatan capaian pertanyaan secara berhierarki (HQR) dan EHQR. Namun begitu, hasil keputusan daripada eksperimen yang dijalankan menunjukkan bahawa pendekatan yang dikemukakan telah dapat mengurangkan jumlah agen-agen mobil dan juga memperbaiki jumlah masa yang digunakan dalam mendapatkan hasil pertanyaan berbanding dengan pendekatan-pendekatan lain.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

1.1 Introduction

Mobile agent is an emerging technology that is gaining momentum in the field of distributed computing. There are some advantages in using mobile agent technology compared with traditional client-server solutions. For example, it can reduce network traffic, it can support a large scale of computations with many computers in a distributed environment, it allows the use of disconnected computing for processing a user queries, and it provides more flexibility in the development and maintenance of distributed applications. The goal of this thesis is based on the application of mobile agent technology in supporting the query retrieval process from the World Wide Web (WWW). Specifically, the methods of dispatching the mobile agents to retrieve the query results from the search engines in WWW have been investigated.

1.2 General Problem Statements

Mobile agents often have a task to collect data from several predefined sites. This should be done in an efficient way by minimizing the elapsed time. Usually these agents only know the list of sites but not the distances between them. This research proposes a method to minimize a network routing time taken by the mobile agents using genetic algorithm (GA) to collect query results from different sites. The mobile agents repeat traveling over short routes and avoid longer ones. The GA will be used to select the best routes that will be used by mobile agents to retrieve image datasets in a very short time. The adaptations of mobile agents for network routing have been investigated by many researchers. For example, Sum et al. [1] and Kinoshita et al. [2]

have proposed a method to reduce the number of agents to be used to retrieve the information from then internet that will minimize the routing time taken by the mobile agents. Chen et al. [3] have proposed a method to reduce the cost and network traffic used by mobile agents to retrieve information from the internet, namely, the highest probability first search (HPFS) algorithm. However, there is a problem with the HPSF where the control function of agents after the target object has been located has not been investigated.

1.3 Objectives

Below are the objectives for this research:

a) To study the available algorithms for mobile agent to select a suitable route to retrieve query results from a designated search engines in the World Wide Web (WWW).
b) To analyze the performance of genetic algorithm (GA) for retrieving the query from the WWW.
c) To analyze the retrieval of queries from distributed databases using different dispatching methods of mobile agent technology.

1.4 Scopes

[1] The scope of this study will focus on the development of a network routing algorithms by mobile agents for query retrieval using genetic algorithm (GA). We will apply the mobile agent technology for image query retrieval from distributed databases across the Web. The mobile agents will be using the shortest routes that have been defined by using the proposed algorithm in order to retrieve the image query results.

[2] A mobile agent search system (MaSS) that has been developed will support the users to retrieve the image query results from the Web. The expected results will be a mobile agents system that enables users to retrieve the information from a distributed database within a minimum time period.

[3] We will compare the effectiveness of the proposed algorithm compared with other techniques such heuristics methods such as an itinerary query retrieval

approach (IQR), a parallel query retrieval approach (PQR), a serial query retrieval approach (SQR), a shuttle query retrieval approach (SHQR), a hierarchical query retrieval approach (HQR) and the EHQR approach. Experimental results will be used to show that the proposed approach reduces the number of mobile agents and also improve the total time taken to retrieve the query results compared with other approaches.

CHAPTER 2

LITERATURE REVIEWS

2.1 Introduction

The introduction of mobile agent in the field of distributed computing has proven useful where the mobile agents will roam the networks to search for information requested by the users. The mobile agents will also cooperate with other agents to accomplish the assigned tasks. There are four main properties belonging to mobile agents such as intelligence, communication, autonomy, and mobility. Maes et al., (1997) have described each of these properties. The biological insects have inspired most of the research related to agent based network routing and their colonies as described by Schoonderwoerd et al., (1997). It relies on the principles that individual insects will perform a simple behavior while the collective communities of these insects will perform complex problem solving capabilities. A research has been conducted in mapping the biological insects to the network routing management by using mobile agents. These agents are represented as artificial agents that traverse the network to collect specific information from the designated hosts. They will visit these hosts and coordinate with other agents to accomplish the assign tasks on behalf of users. They will also make several decisions to adapt their behavior according to the current environment in which they are currently resided. In this chapter, we present several literature reviews related to our research.

2.2 Related Works on Mobile Agents for Query Retrieval

Many researchers have investigated the adaptations of mobile agents for network routing. For example, [12] have proposed a method to reduce the number of agents to be used to retrieve the information from then internet that will minimize the routing time taken by the mobile agents. [10] have proposed an adaptive routing algorithm for network routing by using a rule-based method. [11] have proposed a method to reduce the cost and network traffic used by mobile agents to retrieve information from the internet, namely, the highest probability first search (HPFS) algorithm. The HPSF has been used to locate the agent that has been dispatched to the Internet to collect the required information. However, there is a problem with the HPSF where the control function of agents after the target object has been located has not been discussed.

Dikaiakos et al. [13] have developed a framework for benchmarking the performance of mobile agent systems for the distributed databases. The parameters that have been used in the benchmarks are the types of operating systems (e.g., Unix, Windows, etc.), the channel configuration (e.g., LAN, WAN, etc.), the size of mobile agents and together with the message size, and the loop size which determines the number of times the benchmarks are executed.

Furthermore, Kawamura et al. [14] have analyzed three basic agent paradigms applied to the process of accessing data from distributed databases such as direct access, stationery agent access, and mobile agent access. Also, Papastavrou et al. [15] and Ismail et al. [16] have analyzed the comparisons between a Java applet-based approach and a mobile agent technology in accessing distributed databases from the WWW. However, the Java applet-based approach has a limitation in flexibility, scalability, and robustness in accessing the databases compared with the mobile agent technology.

Moreover, the usage of mobile agents for information filtering has been analyzed by Theilmann et al. [17]. Two deployments approaches applied on mobile agents for retrieving the information such as a basic dissemination algorithm and a hierarchical dissemination algorithm have been proposed. The former is related to the increment of the number of agents to be dispatched to the servers in order to retrieve query results. The latter is related to the process of traversing the mobile agents to the remote sites in order to retrieve the query results. However, the process of dispatching mobile agents to the remote hosts has not been clearly stated by all of the above authors.

Jha et al. [18] have analyzed the quantitative comparisons between mobile agent and client–server approach. However, the mobility patterns of mobile agents are only based on the simple approaches such as a sequential client–server approach, a sequential mobile agent approach, a parallel client–server approach and a parallel mobile agent approach. Also, Cogan et al. [19] have analyzed the performance of mobile agents based on different platforms that are Voyager mobile agent platform [20] and Java Remote Method Invocation (Java-RMI) platform [21]. However, from the experiments, only the mobility and remote method invocation on mobile agents applications have been compared. Also, the parameters such as the number of agents to be dispatched to the hosts that affect the performance of the applications have not been investigated.

Rubinstein et al. [22] have analyzed the performance of mobile agent in network management and comparing it with the client–server model where the topologies are similar to the Internet architecture have been used to analyze the mobile agents technology in a network environment. The results show that the mobile agents approach provides better response time in retrieving query results compared with client–server architecture.

The application of mobile agents in workflow systems in order to improve network performance and scalability has been analyzed by Yoo et al. [23]. Here, a stochastic Petri-nets [24] simulation through the comparison of client–server architecture has been analyzed. However, the comparisons of the effectiveness of the systems have not been clearly identified by the authors.

A framework for building a reusable mobile agent from two kinds of components such as an itinerary component and an application-specific component has been proposed by Satoh [25]. The former is a carrier of the latter over particular networks and the latter defines management tasks performed at each host independently of the network. This framework also provides a mechanism for matchmaking the two mobile-agent based components. Since the mechanism is formulated based on a process algebra approach, it can strictly select a suitable itinerary component to perform management tasks at the hosts that the tasks want to visit over networks. However, the empirical comparisons of the effectiveness of the mobile agents performance have not been clearly stated.

Schulze et al. [26] have proposed new mobile agent architecture with an availability service and a transparency interface. The mobility of agents is based on the proactive and reactive approaches where the former is related to the property of agents that are able to decide when and where to move. The later approach is based on the ability of the agents that will react in consequence of changes in the environment. Onishi et al. [27] have described a new ad hoc routing algorithm, which inspired by multiple entries for each of destinations in the routing table to store much more information from agents and evaluating them to make better use of information, which succeeded in raising network connectivity has been proposed. However, the numbers of mobile agents to be dispatched to the hosts to collect information have not been discussed.

Here, we propose a different method of dispatching mobile agents to the remote hosts in order to retrieve the query results. We try to minimize the number of agents used to fetch the query results from the remote hosts in order to reduce network overload. It is different compared with the previous approaches [14], [15], [15], [17], [18], [19], [22], [23], [25], [26] and [27] by utilizing the mobile agent properties such as object cloning, dispatching, and disposing in the EHQR approach. This is done by combining a hierarchical and a parallel dispatching of mobile agents to the local prefetch servers in order to retrieve the query results in the Mobile Agent Search System (MaSS). Also, we have analyzed the on-line and off-line query retrieval approaches applied by mobile agents in the MaSS as discussed in chapter 3.

In this research, we proposed a network routing by mobile agents for query retrieval using a genetic algorithm (GA). The agent repeats traveling over short routes and avoids longer ones. The GA approach for selecting the best routes has been applied to the mobile agents technology. We have applied the route selection of query retrieval by mobile agents in the MaSS [28]. The result shows that the proposed method provides good time minimization in retrieving the query results by the mobile agents based on different GA parameters.

2.3 Network Routing by Mobile Agents using GA

In order to retrieve the query results in an optimal time, the MaSS search agent has applied the GA approach for route selection in order to minimize the query retrieval time as shown in figure 1. Further formulation of the route optimization is as follows:

$$Min\ Qrt(route)\ \ s.t.\ Delay(route)\ \leq MaxDelay \qquad (1)$$

where *Qrt(route)* is the query retrieval time taken to retrieve the query results, *route* is the paths that have been used by the MaSS search agent **(Y)** to send and retrieve the results from the LP servers. *Delay(route)* is the time constraint due to network bottleneck. The operational process of the MaSS search agent in using GA for selecting an optimal route is shown in figure1. *MaxDelay* is the maximum time delay applied to the path used by the MaSS search agent to retrieve the query results.

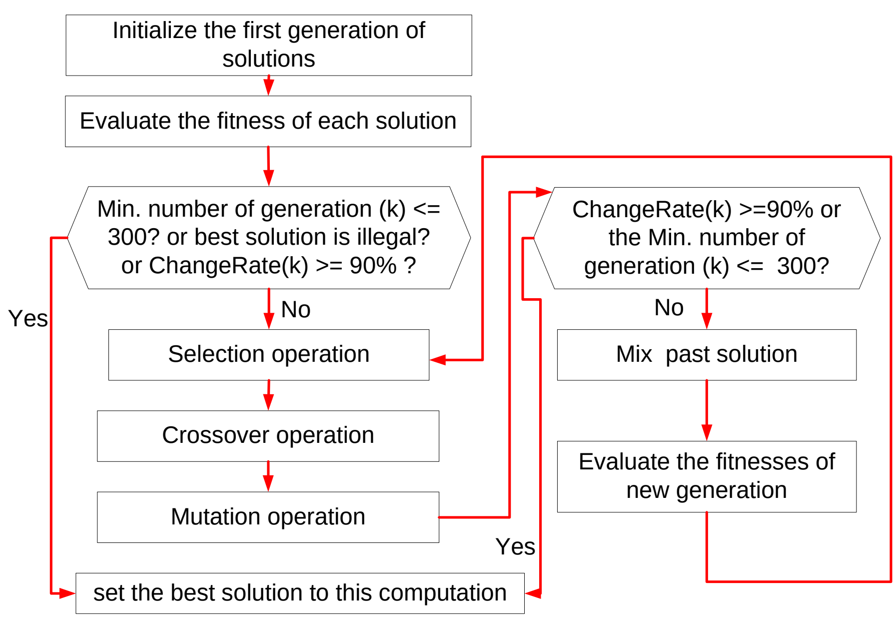

Figure 1 The GA used to select a suitable route by the MaSS search agent.

2.4 Encoding of the GA

The encoding of the GA that has been used by the mobile agents to select the optimal route in retrieving the query results is shown in figure1. We choose path representation approach to encode a route due to its easy implementation. There are 15 LP servers including with the web proxy server as shown in figure 1. The route from the MaSS Server to the web proxy server can be represented as [1 2 3 4 5 6 7 8 9 10 11 12 13 14 15]. If the route does not exist then we include a value of 0 to the route such as [1 2 3 4 5 6 7 8 9 10 0 0 0 0 0].

2.5 Population initialization

We can randomly determine how many nodes the route will pass through and randomly determine which node will be in the route and the sequence of nodes of the route. However there will be some solutions that may violate constraint of delay connectivity. We use a penalty method to deal with these constraints. For those routes that do not exist, we assign a very large delay value to them. For those routes that violate the delay constraint, we add a penalty to their cost. In our algorithm, we use the following expression to evaluate the weighted cost of those illegal routes. It is given by

$$Qrt(route) = Cost(route) + (\alpha + Delay(route)) \qquad (2)$$

where *Qrt(route)* is the weighted query retrieval time for a selected route as described at the beginning of this section, α is the penalty constraint if the route does not exist, e.g., α=1,2,3,…,10, and *Cost(route)* is the function that evaluates the total cost of the links that the route may pass through. The details of *Cost(route)* and *Delay(route)* are shown in tables 1 and 2, respectively.

Table 1 Cost matrix of simulation.

Host's ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	99	15	99	99	99	23	99	99	99	99	99	99	99	99	99
2	15	18	99	99	23	99	99	99	99	99	99	99	99	99	99
3	99	99	15	99	99	99	99	300	99	99	99	99	99	99	99
4	99	23	99	99	99	50	30	99	99	99	99	99	99	99	99
5	99	23	99	99	99	50	30	99	99	99	99	99	99	99	99
6	23	99	99	99	50	99	20	99	18	99	99	99	99	99	99
7	99	99	99	99	30	20	99	10	99	30	30	99	99	99	99
8	99	99	300	99	99	99	10	99	99	99	15	99	99	99	99
9	99	99	99	99	18	99	99	99	99	25	5	99	99	99	99
10	99	99	99	99	99	99	99	99	25	99	99	99	99	99	99
11	99	99	99	99	99	99	99	99	25	99	99	99	99	99	99
12	99	99	99	99	99	99	99	99	25	99	99	99	99	99	99
13	99	99	99	99	99	99	99	99	25	99	99	99	99	99	99
14	99	99	99	99	99	99	99	99	25	99	99	99	99	99	99

Table 2 Delay matrix simulation.

Host's ID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	999	7	999	999	999	10	999	999	999	5	999	999	999	999	999
2	7	3	999	999	6	999	999	999	999	999	999	999	999	999	999
3	999	999	1	999	999	999	999	5	999	999	999	999	999	999	999
4	999	999	999	2	999	999	999	999	999	10	999	999	999	999	999
5	999	6	999	999	999	20	5	999	999	999	999	999	999	999	999
6	10	999	999	999	20	999	20	999	7	999	999	999	999	999	999
7	999	999	999	999	5	20	999	7	999	9	5	999	999	999	999
8	999	999	5	999	999	999	7	999	999	999	1	999	999	999	999
9	999	999	999	999	999	7	999	999	999	3	2	999	999	999	999
10	5	999	999	999	999	999	999	999	3	999	999	999	999	999	999
11	5	999	999	999	999	999	999	999	3	999	999	999	999	999	999
12	5	999	999	999	999	999	999	999	3	999	999	999	999	999	999
13	5	999	999	999	999	999	999	999	3	999	999	999	999	999	999
14	5	999	999	999	999	999	999	999	3	999	999	999	999	999	999

2.6 Fitness Evaluation

Fitness of the solutions is proportional to the chromosomes survivability during the GA operation where the good values are selected and the bad values are discarded. In our case, we have normalized the fitness of solutions to $0 \leq Fitness\ (route) \leq 1$ by the following expression:

$$Fitness\,(route) = \frac{Costs\,(route)}{TotalCosts\,(route) + MaxDelay} \quad (3)$$

where the *Costs(routes)* has been described in previous paragraph and the *TotalCosts(routes)* is the sum of *Costs(routes)* for all populations at generation *k*. It is given by

$$TotalCosts(route) = \sum_{i=1}^{i=k} Costs(route)\ . \qquad (4)$$

For the first 30 generations, we have adjusted the *Fitness (route)* value when the *Fitness (route)* $\leq$ 0.005 to *Fitness (route)* = 0.005 to prevent premature efficiency.

2.7 Selection Operation

In order to keep the "good" solutions and discard the "bad" solutions at the same time, two selection operators have been use in our algorithm. First, we add the fitness of all solutions by randomly generate number between zero and the total value of fitness. Second, we add the fitness value on each of the solutions until the value is greater than the current fitness value of each solution. Then the highest fitness value among the solutions will be selected.

2.8 Crossover Operation

In this research the traditional one-point crossover method is used. Firstly, we find a certain point of the array and swap the part before and after the cross point to generate two new solutions. However, because of the number of nodes the routes may pass through is not fixed, it is difficult to determine a fixed crossover point. So, we assigned a new dynamic crossover point method such as [((A+B)/4)] where A and B are the number of nodes that the two routes will pass through; respectively, and the operator "[]" is the rounding function. For example, two routes before the crossover operations are [1 2 3 4 5 0 0 0 0 0 0 0 0 0] and [6 7 8 0 0 0 0 0 0 0 0 0 0 0]. According to this procedure [((A+B)/4)] after crossover, we get [1 2 8 0 0 0 0 0 0 0 0 0 0 0] and [6 7 3 4 5 0 0 0 0 0 0 0 0 0]. In our approach, only part of the population will exercise the crossover operation.

2.9 Mutation Operation

We have randomly chosen a solution in the population and then change the solution slightly to generate a new solution. In this way, we have chances to find better solution that cannot be found by only crossover operation.

2.10 Repair Operation

During crossover and mutation operations, illegal representation of route may be generated because duplicated elements (node) may appear in the same route. In our algorithm, we delete those duplicated nodes that bring high cost to the route.

2.11 Finding route efficiently and dynamically

Although GA can be used to search in large solution space and obtain an optimal solution, it may take a lot of time to coverage to the optimal solution. In some cases, GA can only find some other sub optimal solution. In practice, we usually want to find a sub optimal solution, which, however, is close to the optimal one. So, in this approach, we have used a combination of conditions to determine when to stop our algorithm's computation. The basic idea is as follows:

After a minimum number of k generations (i.e., $k \leq 300$), if the algorithm has found a feasible solution and has made no improvement for a specific period of time, we will have it stopped. In our algorithm, the "improvement" is presented by the coverage cost rate of the best solution of certain generation. This change rate is evaluated as follows,

$$ChangeRate\,(k) = \frac{Costs\,(k-1)}{Costs\,(k)} \times 100 \qquad (5)$$

where we assume the cost of the best solution of that generation changes at k-th step and *ChangeRate(k)* is the average change rate of cost at k-th step. Once this value is greater than a certain value (i.e., *ChangeRate(k)* ≤ 90%), then we may stop the GA computation. The flow of the GA process for a route selection by the MaSS search agent applied in the MaSS is shown in figure 2.

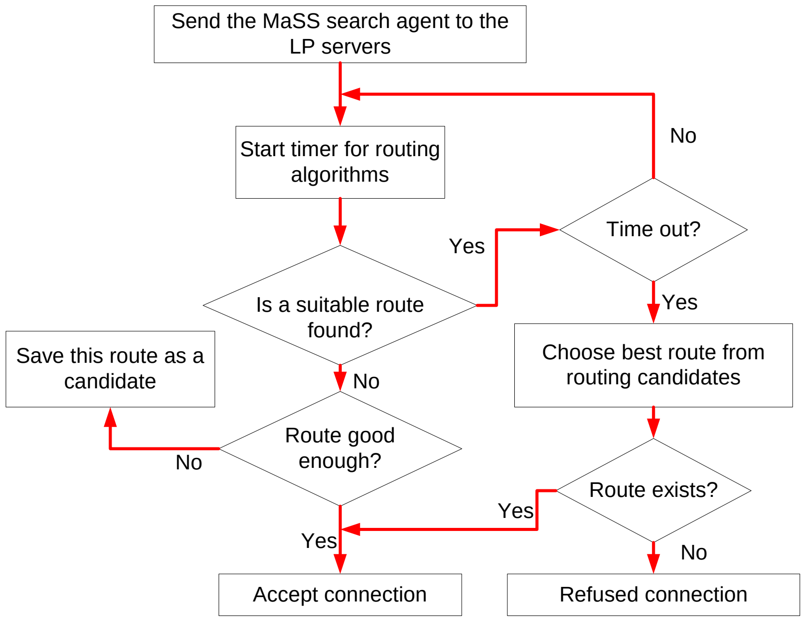

Figure 2 The algorithm that has been used to select a suitable route by the MaSS search agent in order to collect the query results from the LP servers.

2.12 Conclusion

In this chapter, we have discussed about genetic algorithm (GA) which are applied to a mobile agent for query retrieval in the MaSS. Our mobile agent for routing algorithm tries to minimize the query retrieval cost while maintaining a reasonable path delay. The number of generations required to reach a good solution has been reduced significantly by preferring shorter routes in initializing the chromosome pool and reusing the past solutions as the initial chromosomes for the new search.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In this chapter, we discussed the process flow and the architecture of this research. There are several things we focus on to make sure the research completed successfully.

3.2 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 figure 3. The MaSS architecture consists of a MaSS client, a MaSS server, and a collection of the local prefetch (LP) servers. There are two types of query retrieval approaches applied by mobile agents in the MaSS. They are the off-line and on-line approaches. The details architectures of the off-line and on-line retrievals using mobile agents in the MaSS is shown in figure 4 and figure 5. Further descriptions on each of the MaSS components and the query retrieval approaches are described as follows:

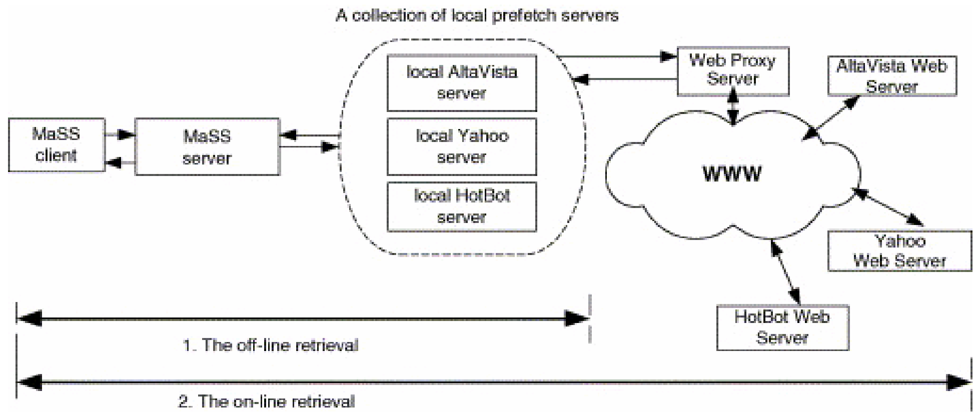

Figure 3 The Mobile Agent Search System (MaSS) architecture.

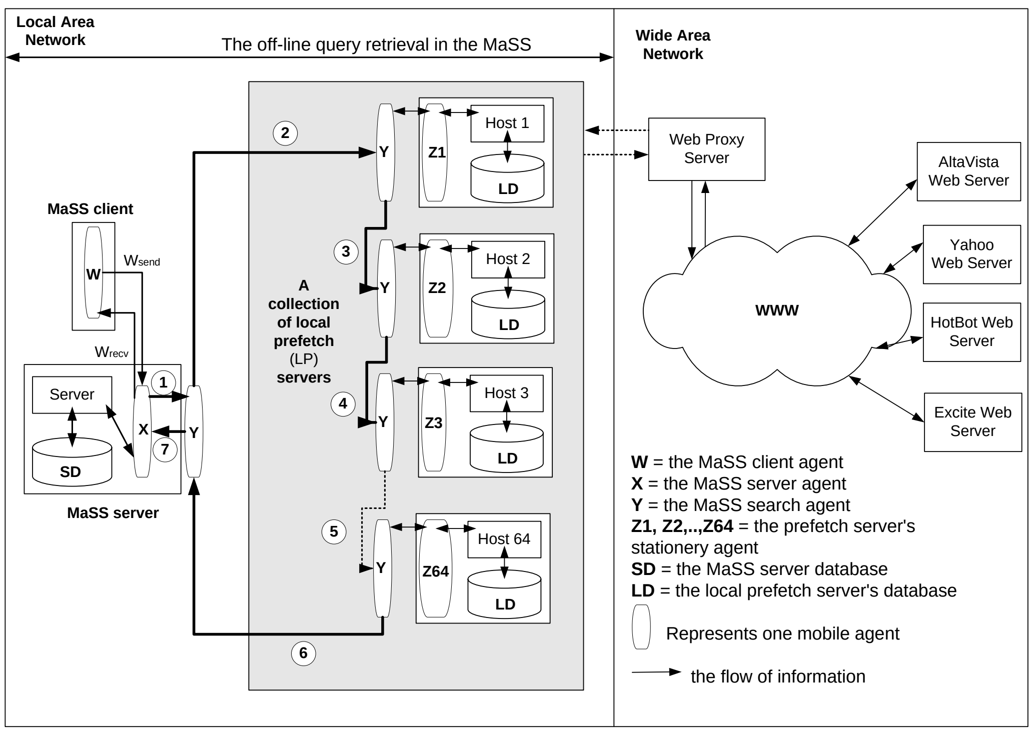

Figure 4 The query retrieval process by the MaSS search agent (**Y**) using the off-line MaSS with 64 local prefetch (LP) servers.

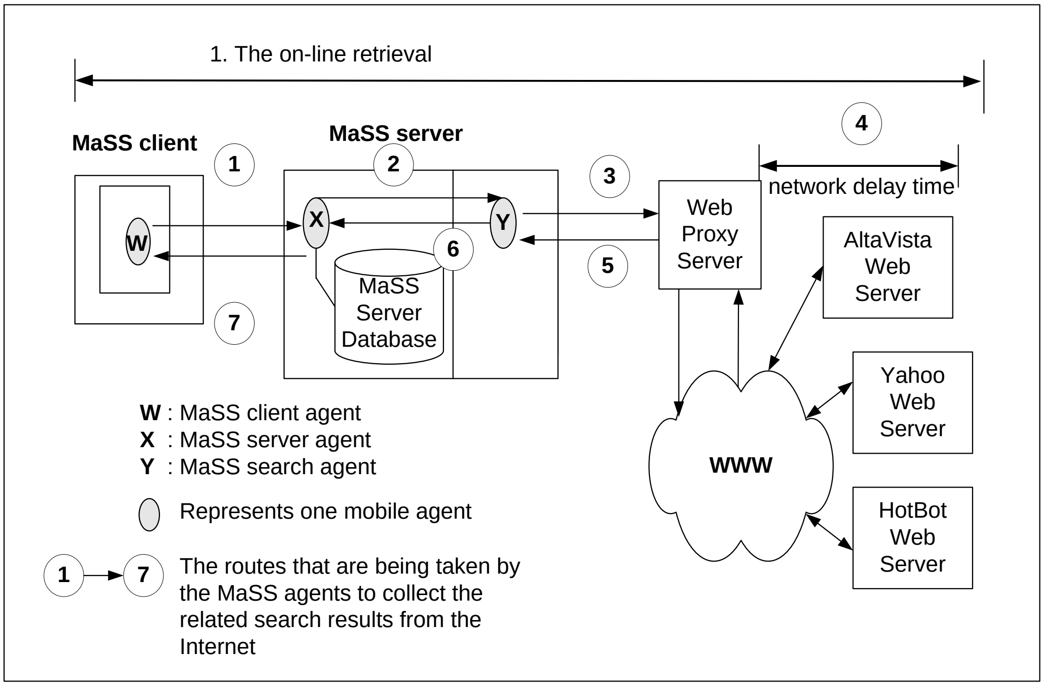

Figure 5 The implementation of mobile agent for the on-line query retrieval approach in the MaSS.

3.3 The MaSS client

The MaSS client consists 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.

3.3.1 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 figure 4 and figure 5.

3.4 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 figure 4 and figure 5. Further descriptions on each of them are as follows:

3.4.1 The MaSS server agent

The MaSS server agent is a stationery agent. Upon receiving request from the MaSS client agent, 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 a Number of Relevant Ordering Score (NROS) method [28]. The quality of query results retrieved by mobile agents in the MaSS using the NROS method has been discussed in [28]. The search results will be stored into the MaSS server database (**SD**) before returning them to the MaSS client agent to be presented to the user as shown in figure 4 and figure 5.

3.4.2 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 LP servers as shown in figure 4. Then it will communicate with the local agent (**Z1**) at each of N hosts where $N = 1, 2, \ldots, H$. In figure 4, the maximum number of hosts H is 64. At host 1, 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 2 and 3 until host 64 with different local agents (**Z1**, **Z2**, **Z3**, …, **Z64**). Once the tasks have been completed, the MaSS search agent will return home and pass the search results to the MaSS server agent.

3.5 The off-line query retrieval process in the MaSS

The off-line query retrieval process by the mobile agents in the MaSS is presented in figure 4. When the MaSS server agent (**X**) receives a request from the MaSS client agent (**W**), it will pass the query item to the MaSS search agent (**Y**). The MaSS search agent will start its routes from the MaSS server to host 1 to collect the query results at the local prefetch server's databases (**LD**). At host 1, the MaSS search agent will ask the local agent (**Z1**) for the search results. Once completed, it continues its search to the LP servers at hosts 2 and 3 until host 64. Also the MaSS search agent will communicate with local agents **Z2** and **Z3** until **Z64** in order to get the query results from different hosts. The MaSS search agent will return to the MaSS server and pass the results to the MaSS server agent. The MaSS server agent will rank the search query results and store them in the server database. Then the search results will be sent to the MaSS client agent to be presented to the user. Others off-line query retrieval approaches are described in section 3.7. The routes taken by the MaSS search agent to collect a query results in the off-line approach are represented by an arrow key from ①→⑦ as shown in figure 4.

3.6 The on-line retrieval process in the MaSS

The on-line retrieval process is presented in figure 5. When the MaSS server agent (**X**) accepts a request from the MaSS client agent (**W**), it will pass the query item to the MaSS search agent (**Y**). Instead of looking for the prefetch servers, the MaSS search agent will search directly from the WWW and retrieve the search results from the search engines. The search results will be stored into the MaSS server database. Here, the processes of removing duplicated and broken links, stemming, stopping, and ranking the search results will take place as shown in figure 6. The WebL script programming language from Compaq [29] to develop a web-fetching program. The script program will fetch the HTML files from the WWW. The results will be sent to the user using an Aglet mobile agent [30].

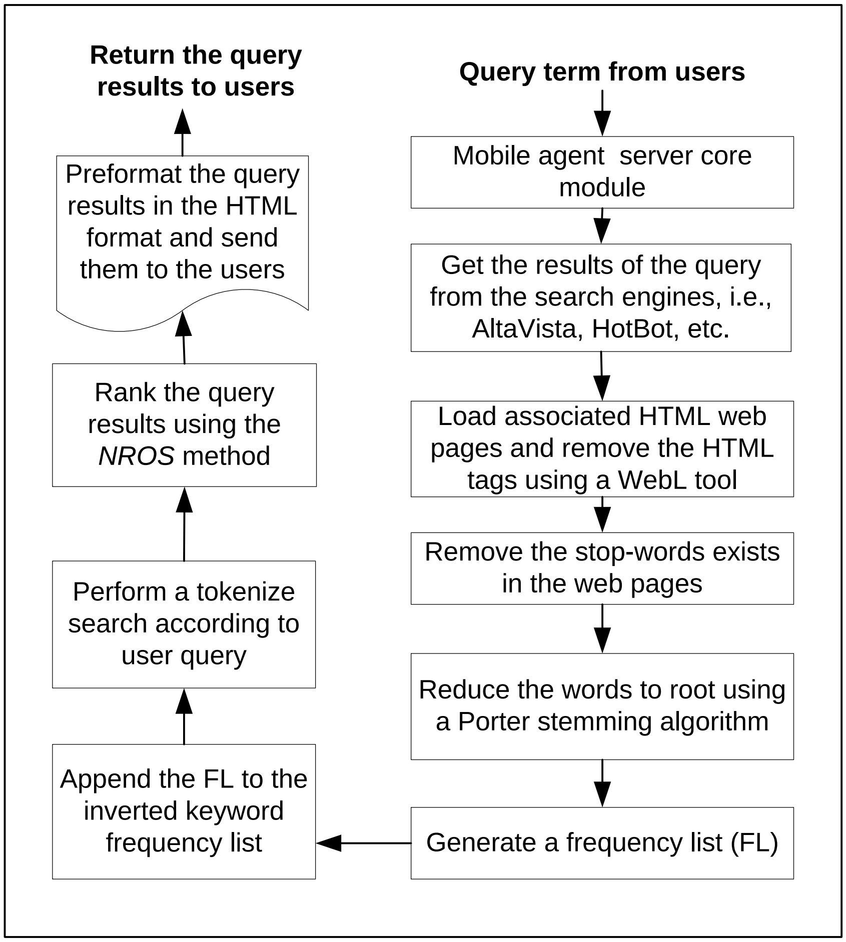

Figure 6 The ranking process of query results retrieved by the mobile agents using the on-line and off-line approaches in the MaSS. The detail of the NROS method has been discussed in Selamat et al. [13].

Referring to figure 6, we have used the stop-words to remove the most frequent word that exists in a web page document such as ‘to’, ‘and’, ‘it’, etc. Removing these words will save spaces for storing document contents and reduce time taken during the search process. Then, a stemming process has been done in order to extract each word from a web page document by reducing it into a possible root word. For example, the words ‘compares’, ‘compared’, and ‘comparing’ have similar meaning with a word ‘compare’. The Porter stemming algorithm [31] has been used to select only the word ‘compare’ to be used as a root word in a web page document. Then the query results will be ranked using the NROS method [28]. The routes taken by the MaSS search agent to collect a query results in the on-line approach are represented by an arrow key from ①→⑦as shown in figure 5.

The on-line retrieval by mobile agents 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 15 as in Table 3. Also we have set the activation time to 24 h 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.

Table 3 The comparison of the time taken by the MaSS using the on-line and off-line query retrieval approaches based on query number 1–15

Query	**DS (kb)**	**CPU (ms)**	**T1 (ms)**	**T2 (ms)**
1. *Computer engineering*	6078	227,303	229,287	1984
2. *Mercedes Benz, Volvo and Fiat*	5535	380,003	382,924	2921
3. *Visiting Australia and Japan*	4640	677,661	678,242	581
4. *The Japan and European Garden*	4380	456,793	458,045	1252
5. *Cancer or body disease*	5819	341,996	343,814	1818
6. *Flower arrangements, origami, Japanese Culture*	6445	744,275	745,740	1465
7. *Food and Beverages in Osaka*	3652	173,387	173,893	506
8. *Biocomputing and evolutions*	3128	165,294	166,115	821
9. *Visiting German and Europe*	2915	176,812	177,833	1021
10. *Cooking Books*	3251	211,725	213,188	1463
11. *Human Computer Interaction*	2603	173,757	174,263	506
12. *Java*	2890	364,751	366,624	1873
13. *Aglets*	2675	255,463	2,656,184	721
14. *Data Mining and Farming*	2322	720,81	73,242	1161
15. *Distributed Programming*	3669	169,571	170,497	926

Note: Query = Query number; DS = Document size (kb); CPU = CPU time taken for the on-line retrieval process; T1 = Total of downloading time using the on-line retrieval process (ms); T2 = Total of downloading time using the off-line retrieval process (ms).

The CPU time taken for query retrieval process using the off-line retrieval approach is not shown in this table because the time taken to rank the query results using the NROS method [28] is minimum.

3.7 The query retrieval approaches applied to the off-line MaSS

There are six query retrieval approaches that have been applied to the off-line MaSS as shown in figure 7. They are an itinerary query retrieval approach (IQR), a parallel query retrieval approach (PQR), a serial query retrieval approach (SQR), a shuttle query retrieval approach (SHQR), a hierarchical query retrieval approach (HQR) and the EHQR approach. The descriptions on each of them are as follows.

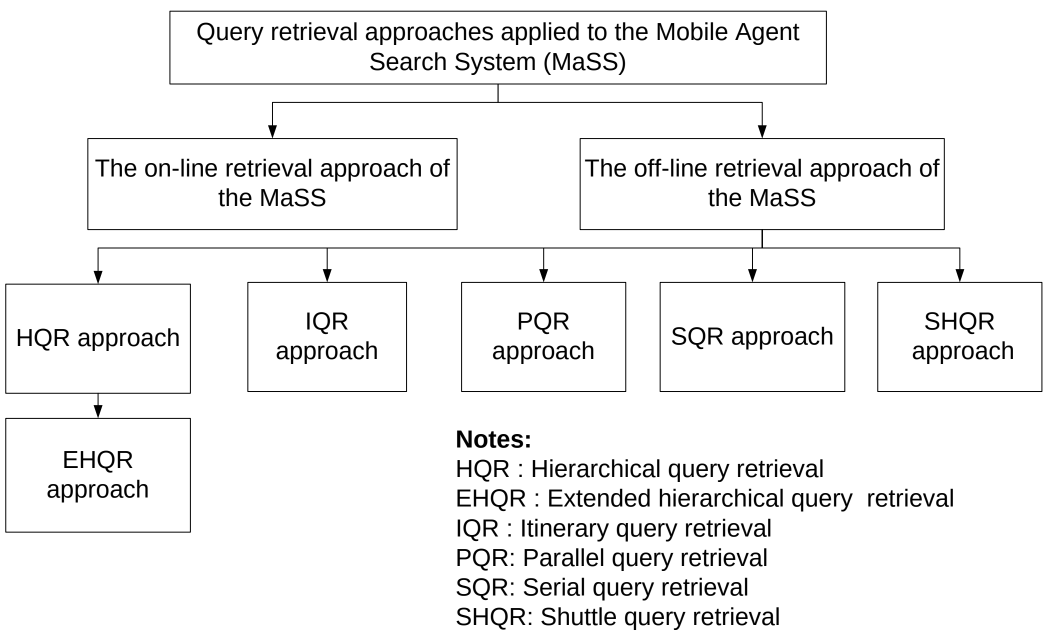

Figure 7 The query retrieval approaches applied to the MaSS.

3.7.1 The IQR approach

In the IQR approach, the MaSS search agent (**Y**) is created by the MaSS server agent (**X**) with the address of a list of LP servers as shown in figure 8. It migrates and visits all of the LP servers in its list one by one. When migrating to the next LP server, it will carry the query results retrieved from the previous and the current LP servers. After it has visited all LP servers, it returns to the MaSS server with the entire retrieved query results. The routes taken by the MaSS search agent to collect the query results by using the IQR approach are represented by an arrow key from ①→⑦ as shown in figure 8.

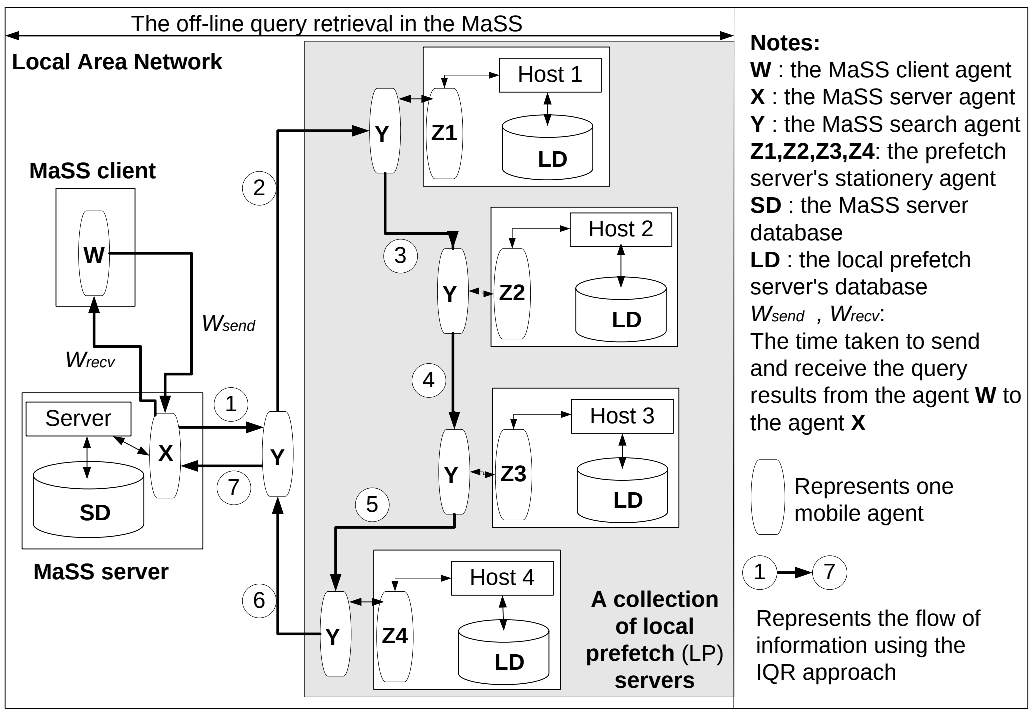

Figure 8 The query retrieval process by the MaSS search agent (**Y**) using itinerary query retrieval (IQR) approach applied to the off-line MaSS.

3.7.2 The PQR approach

In the PQR approach as shown in figure 9, the MaSS server agent (**X**) will dispatch multiple MaSS search agents (**Y**) one by one to all of the LP servers that should be visited. All of the MaSS search agents will access the **LD**s in parallel and send the query results to the MaSS server agent after completing the query process at the **LD** on each of the LP servers. The process of sending each of the MaSS search agents to each of the LP servers is done in serial. The routes taken by the MaSS search agent to collect a query results by using the PQR approach are represented by an arrow key from ①→⑥ as shown in figure 9.

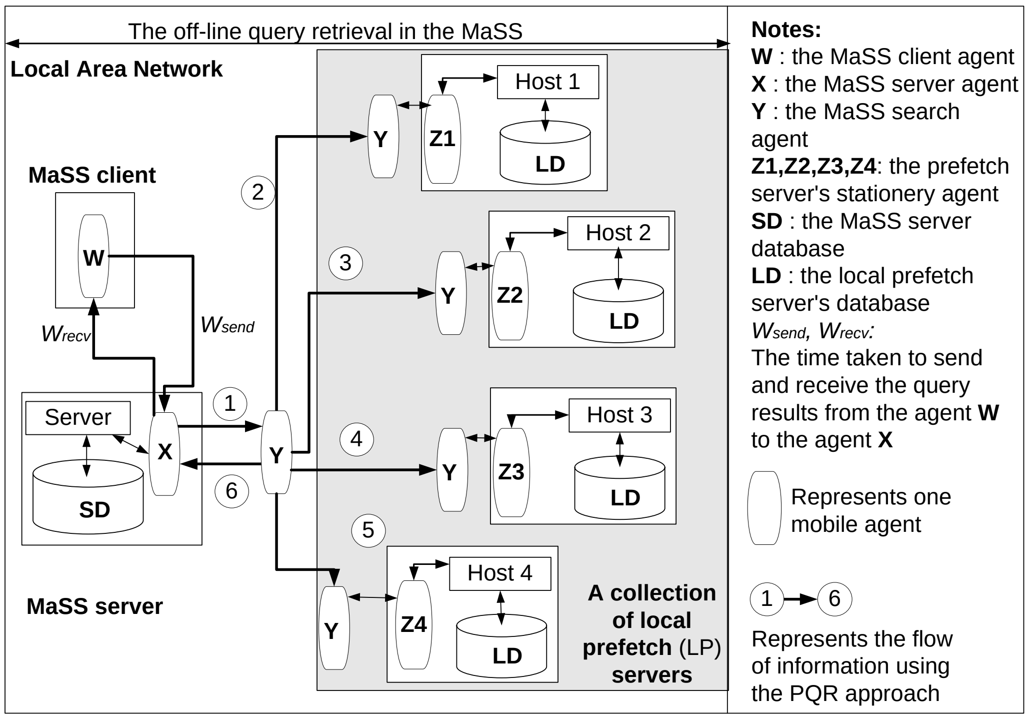

Figure 9 The query retrieval process by the MaSS search agent (**Y**) using the PQR approach.

3.7.3 The SHQR approach

The SHQR approach consists of one MaSS server agent (**X**) and one MaSS search agent (**Y**) as shown in figure 10. The MaSS search agent is dispatched to the LP servers in the list maintained by the MaSS server agent. When the MaSS search agent has read the query results from the **LD** in host 1, it returns to the MaSS server together with the search results. After sending the data to the MaSS server agent, the MaSS search agent will migrate to the next host in the LP servers that is written in the list. The same process is repeated until it arrives at the final host. Then it returns all the query results to the MaSS server agent. The routes taken by the MaSS search agent to collect a query results by using the SHQR approach are represented by an arrow key from ①→⑩ as shown in figure 10.

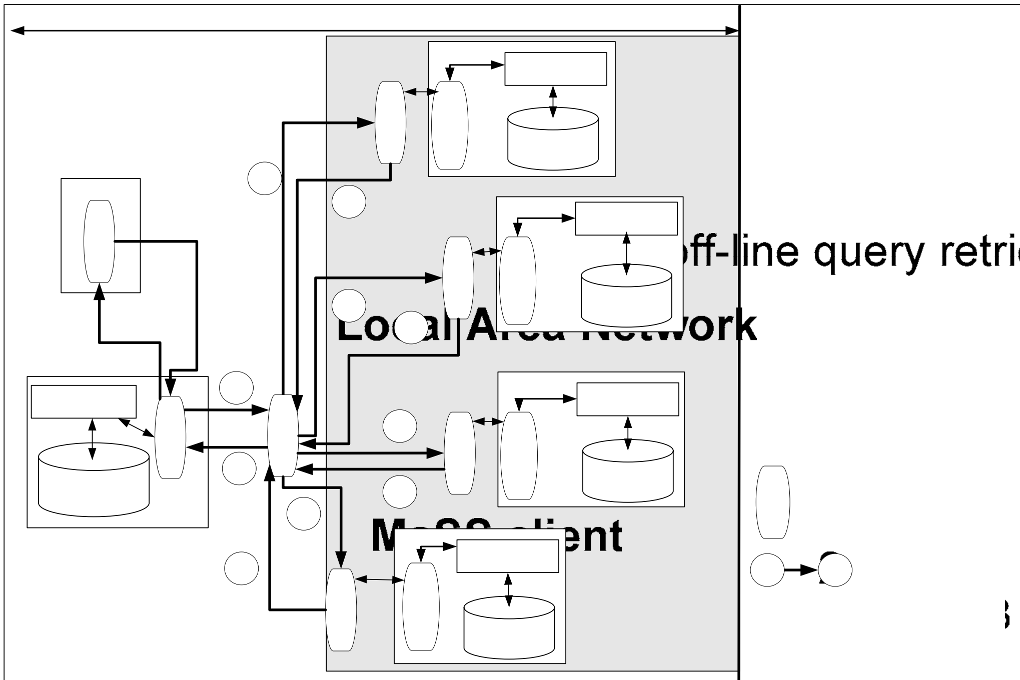

Figure 10 The query retrieval process by the MaSS search agent (**Y**) using the SHQR approach.

3.7.4 The SQR approach

The SQR approach is similar to the SHQR approach. The MaSS server agent (**X**) dispatches an MaSS search agent (**Y**) to the LP server to collect a query results. Upon completing the process, the search agent will return to the MaSS server and pass the query results to the MaSS server agent. The MaSS search agent will be disposed when all of the query results have been given to the MaSS server agent. This is the main different of the SQR approach compared to the SHQR approach as shown in figure 11. The MaSS server agent will create a new MaSS search agent and dispatch it to the next host that has been written in the list. The process is repeated until the MaSS search agent has reached the final host.

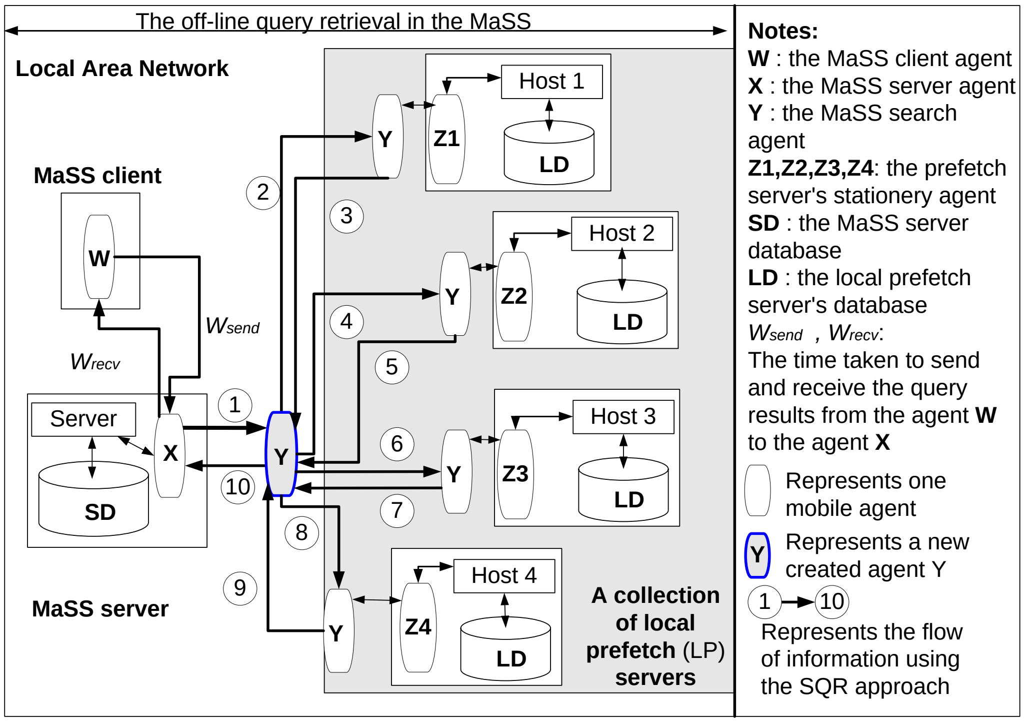

Figure 11 The query retrieval process by the MaSS search agent (**Y**) using the SQR approach.

3.7.5 The HQR approach

The HQR approach is based on a hierarchical structure of the MaSS search agent (**Y**) that will be sent to the LP servers by the MaSS server agent (**X**) as shown in figure 12. The dispatching of the MaSS search agents to the LP servers is done in serial. To simplify the HQR architecture, we represent it as a *tree* as shown in figure 12 and figure 13. Suppose that the MaSS server agent (**X**) is a root of the *tree* which is represented as t_p in figure 13. The *tree* has a height (h = 3) which is represented as h_1, h_2, and h_3 in figure 12. The number of branch (β) is set to 4. The MaSS search agent (**Y**) will be dispatched to a collection of LP servers to search for the query results. In our example, the number of LP servers or hosts H is 64 as described in section 3.4.2.

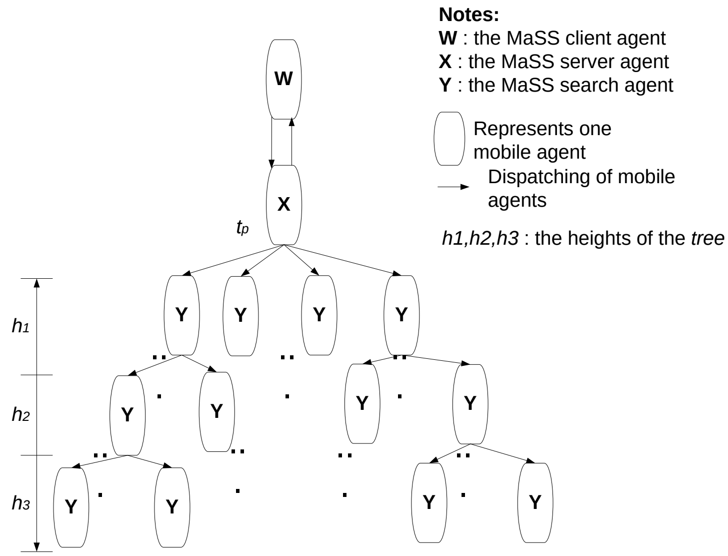

Figure 12 The *tree* architecture of dispatching the MaSS search agent (**Y**) to 64 LP servers using the HQR and EHQR approaches. The height of the *tree* (h) is 3 and the number of branch (β) is 4.

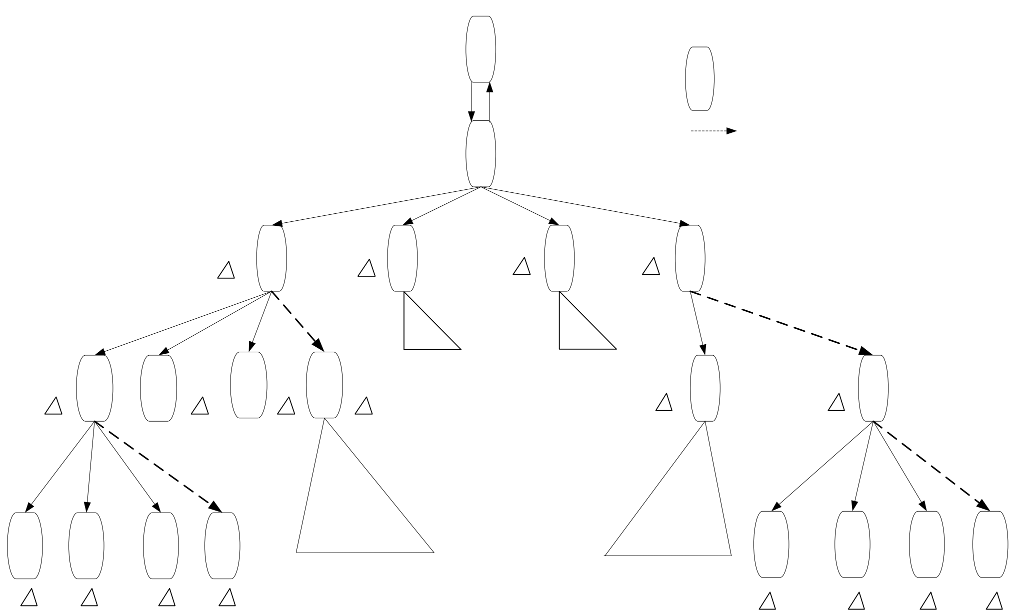

Figure 13 The *tree* architecture of dispatching the MaSS search agent (**Y**) for retrieving the query results using the HQR approach from 64 LP servers. The height of the *tree* (h) is 3 and the number of branch (β) is 4.

Referring to figure 13, Δt is the time to dispatch each of the MaSS search agents to the LP servers. The values of the parameters h, β, and H are based on the number of LP servers to be used in the experiments that will be described in section 3.8.

Referring to figure 13, the MaSS server agent will create and dispatch the MaSS search agents ($t_{1,1}$, $t_{1,2}$, $t_{1,3}$, and $t_{1,4}$) at time Δt, $2\Delta t$, $3\Delta t$, and $4\Delta t$, respectively, to the LP servers 1, 2, 3, and 4. $t_{i,j}$ denotes the MaSS search agent j at the ith level. Next, the parent MaSS search agent ($t_{1,1}$) will create and dispatch another MaSS search agents ($t_{2,1}$, $t_{2,2}$, $t_{2,3}$, and $t_{2,4}$) at time $5\Delta t$, $6\Delta t$, $7\Delta t$, and $8\Delta t$, respectively, to the LP servers 5, 6, 7, and 8. The same process will be repeated until the parent MaSS search agent ($t_{2,16}$) will create and dispatch another MaSS search agents ($t_{3,61}$, $t_{3,62}$, $t_{3,63}$, and $t_{3,64}$) at time $81\Delta t$, $82\Delta t$, $83\Delta t$, and $84\Delta t$, respectively, to the LP servers 61, 62, 63, and 64. The total number of the MaSS search agent to be created by using the HQR approach is 4 + 16 + 64 = 84. Therefore, the total time taken to retrieve the query results using the HQR approach is represented as below:

$$T_{HQR} = W_{send} + W_{recv} + \sum_{\sigma=1}^{\sigma=84} \left(R_\sigma - S_\sigma \right) \tag{1}$$

where the S_σ and R_σ represent the time taken to send and receive the query results by the MaSS search agent j from the LP servers applied in the HQR and EHQR approaches. σ denotes the number of mobile agents will be dispatched to the LP severs by using the HQR approach where $\sigma = 1, 2, 3, \ldots, 84$. Referring to figure 13, the S_σ is given by

$$\underline{S_\sigma = \sigma \Delta t.} \tag{2}$$

The values of S_σ and R_σ can be obtained through experiment that will be explained in section 3.8. The parameters of W_{send} and W_{recv} are the time taken to send and receive the query results from the MaSS client agent (**W**) to the MaSS server agent (**X**) as described in section 2.2.

3.7.6 The EHQR approach

The EHQR and HQR approaches which have been applied to the off-line MaSS are based on the utilization of object cloning, dispatching, and disposing properties that exist in the mobile agent technology [30]. The EHQR approach is based on a hierarchical and a parallel dispatching of the MaSS search agents (**Y**) to the LP servers to retrieve the query results as shown in figure 14. Furthermore, it is based on the combination of the PQR and HQR approaches where the PQR approach has been explained in section 3.7.2. The parameters used in the HQR approach are also used in the EHQR approach.

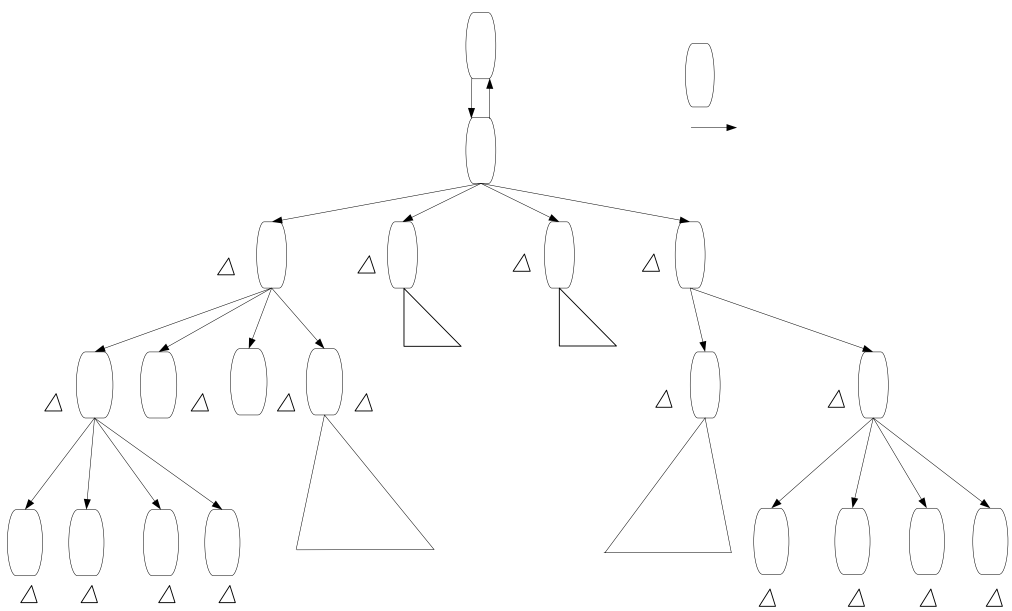

Figure 14 The *tree* architecture of dispatching the MaSS search agent (**Y**) for retrieving the query results using the EHQR approach from 64 LP servers. The height of the *tree* (h) is 3 and the number of branch (β) is 4.

Suppose that the MaSS server agent (**X**) is represented as t_p which is a root of a *tree* as shown in figure 14. It will create and dispatch in parallel the MaSS search agents ($t_{1,1}$, $t_{1,2}$, $t_{1,3}$, and $t_{1,4}$) at time Δt, $2\Delta t$, $3\Delta t$, and $4\Delta t$, respectively, to the LP servers 1, 2, 3, and 4. Then the MaSS search agent ($t_{1,1}$) will create and dispatch another MaSS search agents ($t_{2,1}$, $t_{2,2}$, and $t_{2,3}$) at time $5\Delta t$,$6\Delta t$, and $7\Delta t$, respectively,

W

X

t_p

$t_{1,1}$ $t_{1,2}$

to the LP servers 5, 6, and 7. Upon completing the dispatching of the MaSS search agent ($t_{2,3}$) at time $7\Delta t$, the MaSS search agent ($t_{1,1}$) will dispatch itself to the LP server 8 at time $7\Delta t$. The process of dispatching the MaSS search agents ($t_{2,3}$) and ($t_{1,1}$) is done in parallel.

The same process will be repeated until the MaSS search agent ($t_{1,4}$) will create and dispatch its MaSS search agents ($t_{3,61}$, $t_{3,62}$ and $t_{3,63}$) at time $62\Delta t$, $63\Delta t$, and $64\Delta t$, respectively, to the LP servers 61, 62, and 63. Upon completing the dispatching of the MaSS search agent ($t_{3,63}$) at time $64\Delta t$, the MaSS search agent ($t_{1,4}$) will dispatch itself to the LP server 64. The process of dispatching the MaSS search agents ($t_{3,63}$) and ($t_{1,4}$) is done in parallel at time 64 Δt.

The total number of MaSS search agents to be dispatched to the LP servers is 4 + 12 + 48 = 64. Here, we try to reduce the number of mobile agent that roam the networks because if the numbers of MaSS search agents that will be used to retrieve the query results from the LP servers have been decreased, the network loads will also be reduced and the time taken to retrieve the query results can be minimized.

The total time taken to retrieve the query results using the EHQR approach is represented as below,

$$T_{EHQR} = W_{send} + W_{recv} + \sum_{\sigma=1}^{\sigma=64}(R_\sigma - S_\sigma) \tag{3}$$

where the parameters of W_{send}, W_{recv}, S_σ, and R_σ have been described in the HQR approach. The umber of mobile agents that will be dispatched to the LP servers by using the EHQR approach (σ) is 64.

As the number of mobile agents used to retrieve the query results using the EHQR approach compared with the the HQR approach, we expect that the performance of the MaSS using the EHQR is better than the HQR approach. The simulation results using the HQR and EHQR approaches will be discussed in chapter 4.

3.7.7 The average of roundtrip time

The roundtrip time is the time taken by the MaSS client to send a query to the MaSS server, the MaSS server will send the MaSS search agent to each of the LP servers and return the query results to the MaSS client. The average of roundtrip time on each of the query retrieval approaches will be described in this section. The average of roundtrip time for the IQR approach is represented as below:

$$T_{IQR} = \frac{1}{\tau}\sum_{l=1}^{\tau}\left(W_{send_l} + t(\theta)_l - t(\phi)_l + W_{recv_l}\right) \tag{4}$$

where $t(\theta)$ is the timestamp registered by the MaSS search agent prior to departing from the MaSS server to the LP servers, $t(\phi)$ is the timestamp computed by the MaSS search agent immediately upon returning to the MaSS server from the LP servers, the parameters W_{send} and W_{recv} have been described in section 3.3. The parameters of H and N have been described in section 3.4.

l is the number of trials that have been done in each of the experiments where $l = 1,\ldots,\tau$ and τ is the maximum number of trials. The parameters W_{send}, W_{recv}, l, τ, H, N, $t(\theta)$, and $t(\phi)$ also will be used to calculate the average time taken to retrieve the query results by using the PQR, SQR, and SHQR approaches.

The average of the total time taken for query retrieval for the PQR and SHQR approaches is represented as below:

$$T_{PQR} = T_{SHQR} = \frac{1}{\tau}\sum_{l=1}^{\tau}\left(W_{send_l} + W_{recv_l} + \sum_{N=1}^{H} + \left(t(\theta)_{l,N} - t(\phi)_{l,N}\right)\right) \tag{5}$$

where Eq. (5) is also applied to the SHQR approach. The average of the total time taken for query retrieval using the SQR approach is represented as below:

$$T_{SQR} = \frac{1}{\tau}\sum_{l=1}^{\tau}\left(W_{send_l} + W_{recv_l} + \sum_{N=1}^{H} + \left(t(\theta)_{l,N} - t(\phi)_{l,N} + t_{contect_{l,N}}\right)\right) \tag{6}$$

where t_{context} is the time taken to create a new MaSS search agent upon a disposal of the previous MaSS search agent when it returns home to the MaSS server after the first trip to the LP servers.

3.8 Experiments

3.8.1. The on-line retrieval evaluation

As for the evaluation of the on-line retrieval technique, the MaSS server agent will send users request to the MaSS search agent. It is referring to the total time taken from ①→⑦as shown in figure 4. 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 described in section 3.6. 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 3.6. As these processes are done in a local MaSS server in real-time mode, the time taken to process the query and send the results to a user is significantly long compared to the off-line retrieval as shown in table 3. Also the network delay as in figure 4, contributes the time taken to finish the retrieval process.

3.8.2 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→⑧as shown in figure 5. 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 as described in sections 3.5 and 3.6, respectively. Also, the comparisons between the PQR, HQR, SQR, IQR, SHQR, and EHQR approaches will be evaluated in a different set of experiments.

3.9 Experiment setups

The experiments have been done during the night-time since we did not have a dedicated and isolated network. Performing measurements at night allowed us to minimize the congestion and adverse factors. Our testbed workstations are connected to a 100 Mb/s Ethernet LAN segment. The evaluations on the quality of the query results that have been retrieved by the MaSS search agent (**Y**) are not discussed in this paper. Further information on query ranking and evaluations using the MaSS can be found in Selamat et al. [28]. There are three experiments have been conducted in order to evaluate the performance of query retrieval using the approaches that have been described previously.

In the first experiment, the time taken to retrieve different types of queries by mobile agents is shown in table 3. For this experiment, we only compare the on-line and off-line query retrieval approaches by mobile agents in the MaSS. Further detail experiments are conducted in experiments two and three, respectively.

In the second experiment, the average roundtrips time for the IQR, PQR, SQR, and SHQR approaches are measured using Eqs. (4), (5) and (6). In each approach, the MaSS search agent is dispatched to the LP servers to retrieve 100, 200, 300, 500, and 1024 Kbytes, of query results, respectively. The number of LP servers (H) is set to 4, 8, 12, and 16. These variants aim to observe the performance differences with different experiments and illustrate how these variations will impact the performance. In each of the experiments, we have rebooted the MaSS client host, the MaSS server host, and the LP servers in order to avoid the impact of class cache on the execution time of repeated experiments.

In the third experiment, the comparisons of the time taken for a query retrieval process using the HQR and EHQR approaches have been investigated. The number of LP servers between 4 and 16 hosts did not affect the performance of the query retrieval compared with the first experiment. Therefore, we have increased the number of LP servers (H) to 64. The size of query results is set to 100 Kbytes. Eqs. (1), (2) and (3) have been used to find the total time taken to retrieve the query results using the HQR and EHQR approaches. The summary of the parameters used in the experiments one, two, and three is shown in table 4.

Table 4 The query retrieval approaches applied to the on-line and off-line MaSS

Experiment no.	**QRA**	**MDA**	**LP servers (*H*)**	**X**
Experiment one	Off-line	Serial	16	16
Experiment one	On-line	Serial	–	1
Experiment two	PQR	Parallel	16	16
Experiment two	IQR	Itinerary	16	1
Experiment two	SHQR	Serial	16	1
Experiment two	SQR	Serial*	16	16
Experiment three	HQR	Hierarchical and serial	64	84
Experiment three	EHQR	Hierarchical and parallel	64	64

Notes: QRA: Query retrieval approach; MDA: Mobile agent dispatching approach; LP servers (*H*): The number of LP servers; **X**: The number of the MaSS search agents; Serial* The Mass search agent is created and disposed when arriving at the MaSS server and On-line: The Mass search agent is retrieving query results directly from the search engines in the WWW such as Altavista, Yahoo, Excite, etc.

3.10 Conclusion

In this chapter, we have described the architecture of MaSS. There are several approaches we used in this research from the view of on-line retrieval approach and the off-line approach. For instance, the hierarchical query retrieval, the extended hierarchical query retrieval and itinerary query retrieval for on-line retrieval approach of the MaSS. For the off-line approach of the MaSS, there are parallel query retrieval, serial query retrieval and shuttle query retrieval used for comparison.

CHAPTER 4

DATA AND DISCUSSION

4.1 Introduction

The experiments and the performance parameters of mobile agent using the GA routing have been conducted in a local area network (LAN) as shown in figures 15 and 16. In this figure, we only represent the nodes of interest (i.e., access nodes or edge nodes) and their interconnections. In the following we assume that cost and delay of links between two nodes at certain period time are given. If there are no links exist between nodes or bandwidth of the link cannot support the traffic request between the MaSS server and the web proxy server, a large delay value (i.e., 999) is assigned to that link.

The objective of our experiment is to find the route from the MaSS server to the web proxy server that has the lowest cost and the delay is less than the maximum delay requirement. The cost matrix and the delay matrix are given in tables 1 and 2, respectively. In these tables, element ($i,j,\ i \neq j$), is the Cost(delay) from node i to node j where $i, j \in [1,2,,3,\ldots,15]$, element ($i,i$) is the Cost(delay) from the MaSS server to node i, element (i, 15) is the Cost(delay) from node i to the web proxy server. The upper-bound requirement of time delay (MaxDelay) is assumed to be 25 times in this study. The parameters of the GA are shown in table 5.

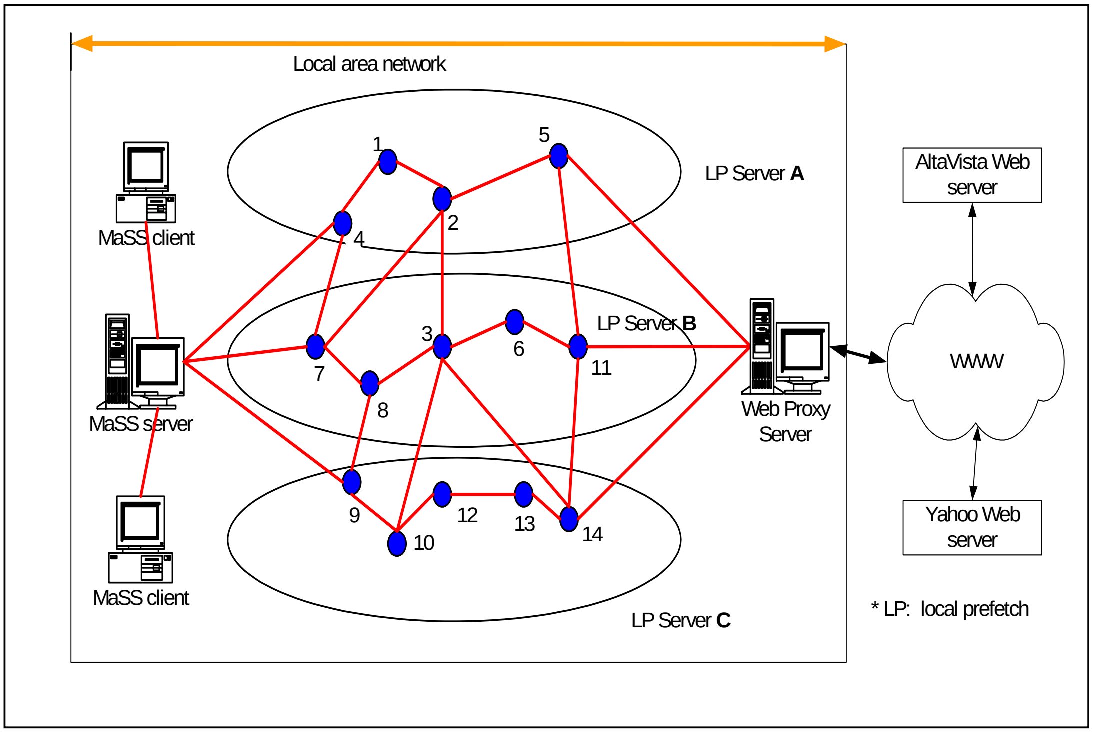

Figure 15 The Mobile Agent Search System (MaSS) architecture.

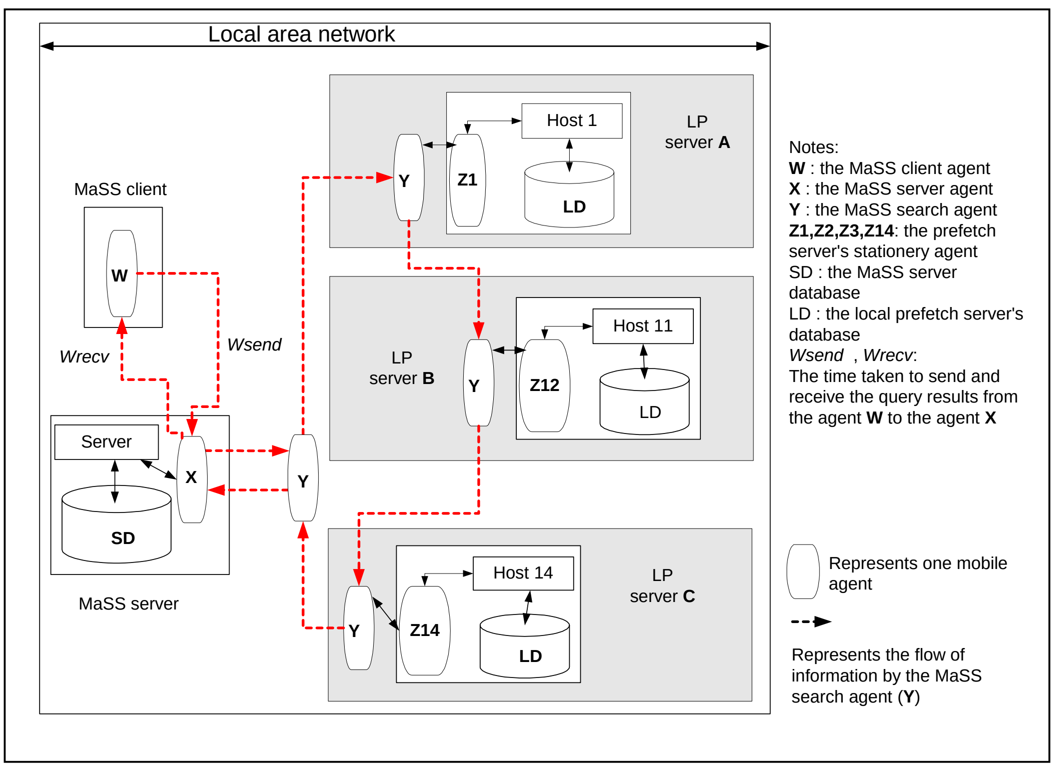

Figure 16 The detail of the MaSS architecture.

Table 5 Genetic algorithm parameters.

Items	Run-1	Run-2	Run-3
No. of populations	320	320	320
No. of genomes	14	14	14
Crossover rate	0.85	0.85	0.85
Mutation rate	0.7	0.5	0.24
No. of generations	420	420	420

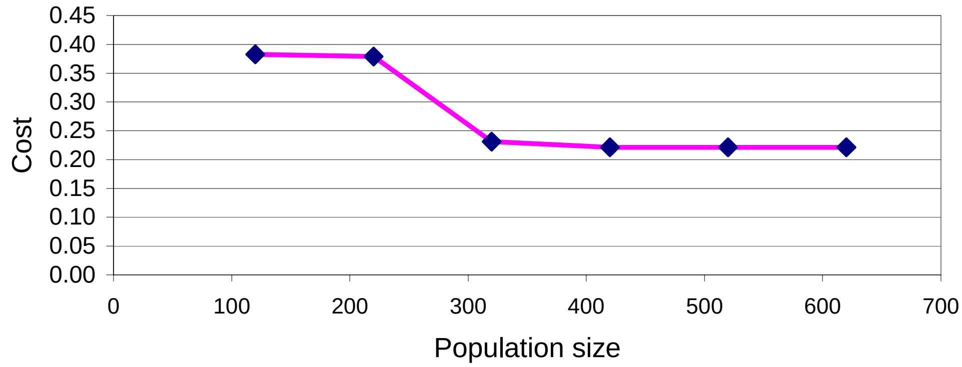

Figure 17 The effect of the size of chromosome pool on the performance of the proposed algorithm.

4.2 Results

The results from each of the experiments are described as follows:

4.2.1 Results for experiment one

The result in experiment one indicates that the mobile agent has provides a significant time saving using the off-line query retrieval approach for sending results from the LP servers to the MaSS client compared with the on-line retrieval approach. The detail of query types is shown in table 3. 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 229,287 ms. Comparatively, by using the off-line retrieval process, the time that the user has to wait until the query results are received is 1984 ms. Also for the query title *Visiting Australia and Japan*, the time taken using the MaSS in the off-line retrieval process is 581 ms. In comparison, the time taken using the on-line retrieval process is 678,242 ms. Further examples of the time taken by user to be on-line and off-line with different queries have been shown in table 3. The average percentage of time reduction by using mobile the off-line query retrieval approach compared with the on-line approach is 95.06%.

4.2.2 Results for experiment two

The results for experiment two are shown in figure 18, figure 19, figure 20, figure 21 and figure 22 are the average of roundtrip time from τ independent trials as explained in section 3.7.7. Figure 18 represents the average time taken to retrieve 100 Kbytes of the query results from the LP servers. It shows that the PQR approach performs better than other approaches when 16 LP servers are used. Also the same characteristic of the PQR approach can be seen in figure 19, figure 20, figure 21 and figure 22 where the sizes of query results are 200, 300, 500, and 1024 Kbytes, respectively. With respect to the SHQR approach, it performs well compared to the SQR model as shown in figure 18 and figure 21 when 16 LP servers are used. However, the time taken by the SHQR approach does not perform well when the sizes of query results are 200, 300, and 1024 Kbytes, respectively, as shown in figure 19, figure 20 and figure 22. Furthermore, the time taken for the IQR approach is the worst when the size of query results and the number of LP servers are increased as shown in figure 19, figure 20, figure 21 and figure 22. However, the PQR approach performs better compared with other approaches in retrieving the query results from the LP servers for all sizes of query results in the second experiment.

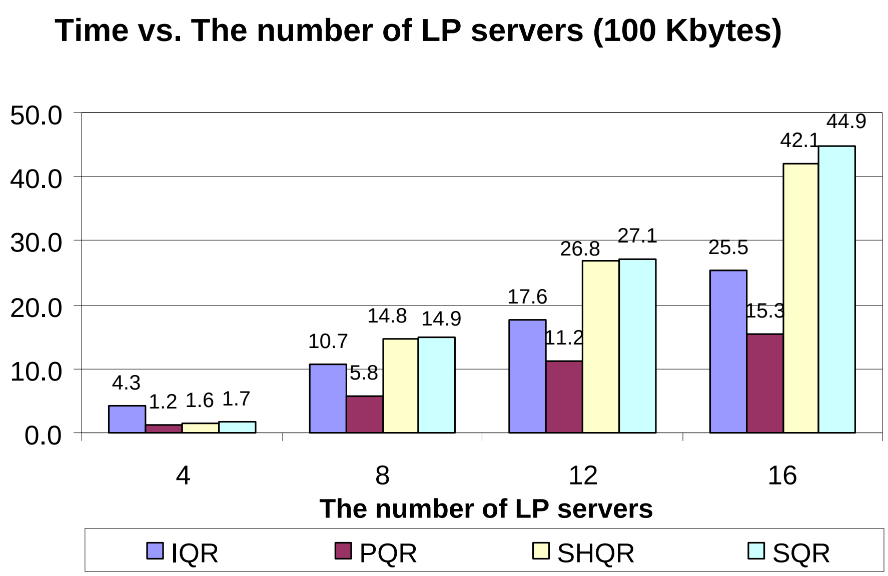

Figure 18 The roundtrip time taken by each of the approaches to retrieve 100 Kbytes of query results from 16 LP servers.

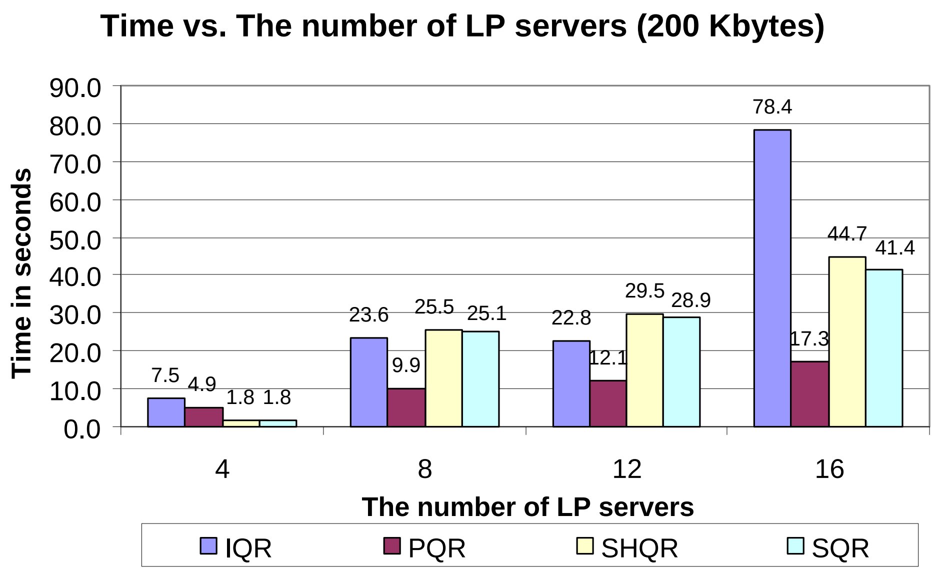

Figure 19 The roundtrip time taken by each of the approaches to retrieve 200 Kbytes of query results from 16 LP servers.

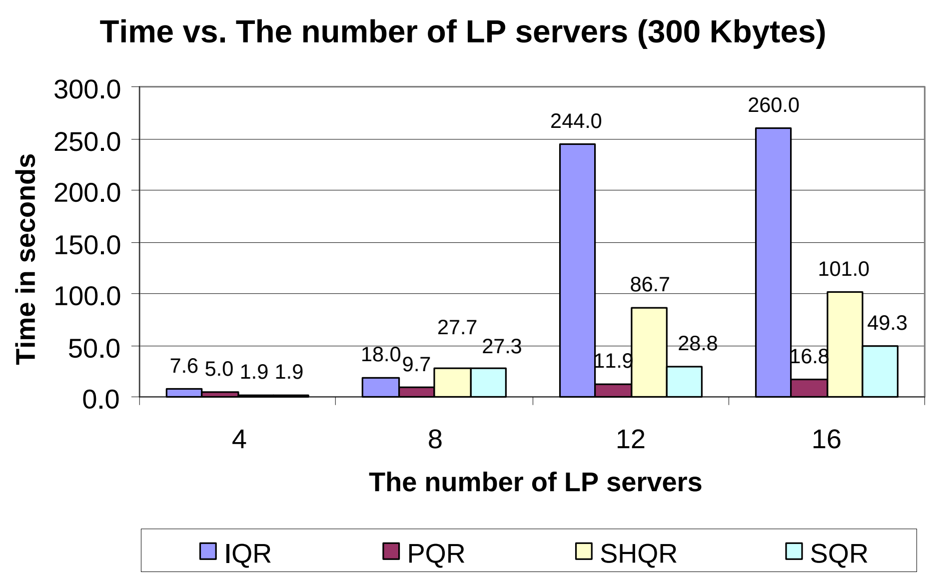

Figure 20 The roundtrip time taken by each of the approaches to retrieve 300 Kbytes of query results from 16 LP servers.

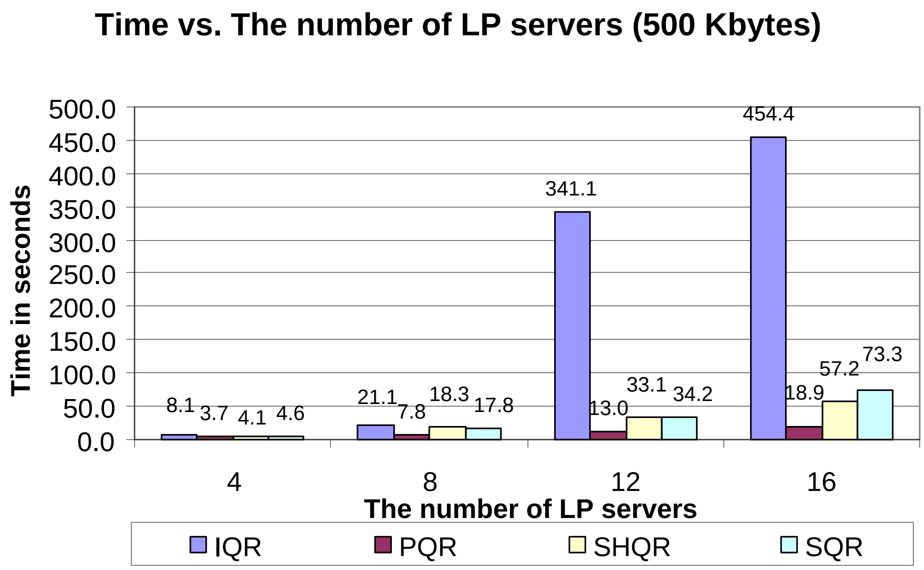

Figure 21 The roundtrip time taken by each of the approaches to retrieve 500 Kbytes of query results from 16 LP servers.

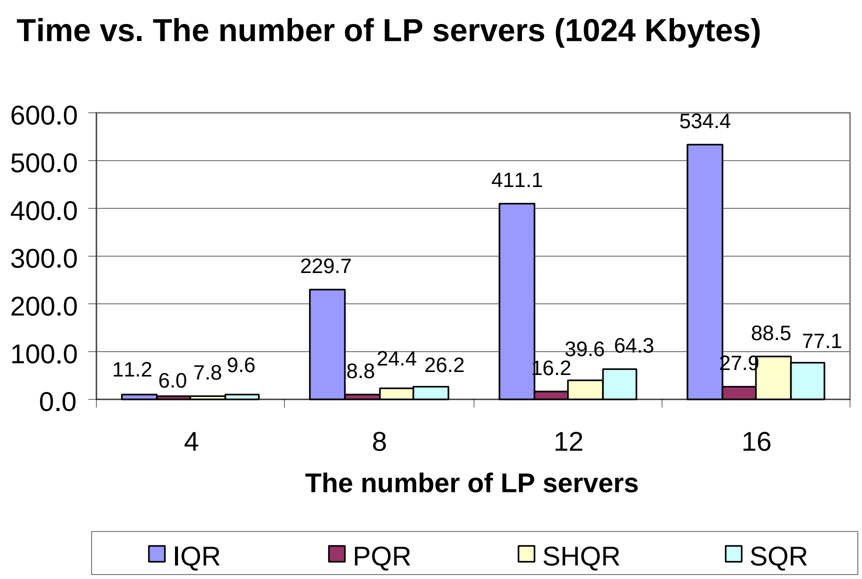

Figure 22 The roundtrip time taken by each of the approaches to retrieve 1024 Kbytes of query results from 16 LP servers.

4.2.3 Results for experiment three

The results of experiment two is shown in figure 23. The performance of the EHQR is better than the HQR approach. For example, the total time taken to retrieve 100 Kbytes query results from 64 LP servers using the HQR approach is 55 s.

However, the time taken to retrieve the same size of query results using the EHQR approach is 35.5 s.

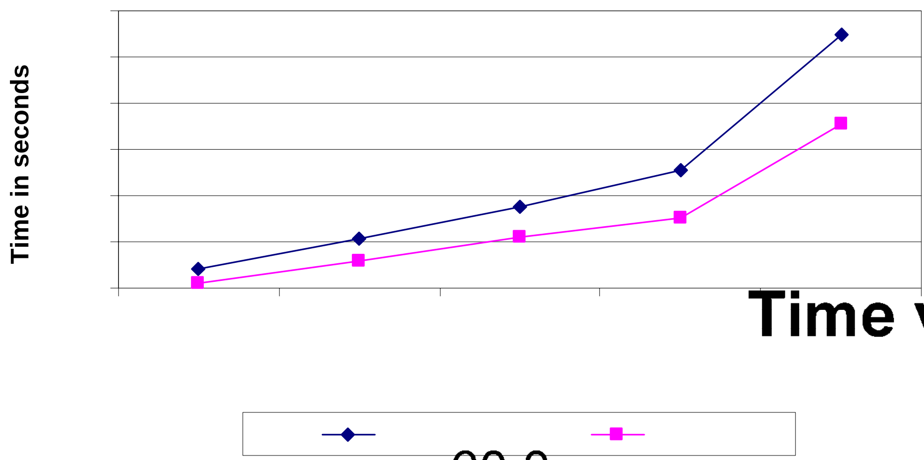

Figure 23 The roundtrip time taken by the EHQR and HQR approaches to retrieve 100 Kbytes query results from 64 LP servers.

4.3 Discussions

The performance of mobile agents for information retrieval has previously been investigated by many research groups as described in Section 2. However, our proposed query retrieval approach is different compared with the previous approaches [14], [15], [16], [18], [19], [22] and [17] by utilizing the mobile agent properties such as object cloning, dispatching, and disposing. This is done by combining a hierarchical and a parallel dispatching of mobile agents to the LP servers in order to retrieve the query results which have been applied in the EHQR approach as described in section 3.7.6. The summaries of the results taken from experiments one, two and three are shown in table 6. The performance of mobile agents for query retrieval in the MaSS has shown that the off-line approach is better than the on-line approach as described in the first experiment.

Table 6 The summary of the results from all of the experiments

Experiment no.	LP servers (*H*)	Retrieval approach	Best approach
Experiment one	16 LP servers	On-line and Off-line	Off-line
Experiment two	16 LP servers	IQR, PQR, SHQR, and SQR	PQR
Experiment three	64 LP servers	HQR and EHQR	EHQR

Notes: LP servers (*H*): The number of LP servers; Retrieval approach: The query retrieval approaches applied to the off-line MaSS and Best approach: The best query retrieval approaches.

The performance of mobile agents using the PQR approach is better compared with the IQR, SQR, and SHQR approaches when we retrieve the query results from the LP servers with different sizes of query results in the second experiment. For example, the roundtrips time taken to retrieve 1024 Kbytes of query results from 16 LP servers using the PQR approach is 27.9 s. However, the roundtrips time taken to retrieve the same size of query results using the IQR, SQR, and SHQR approaches are 534.4, 77.1, and 88.5 s, respectively. Approximately, 94.78%, 63.81%, and 68.47% of the times reductions have been achieved by using the PQR approach compared with the IQR, SQR, and SHQR approaches, respectively, when we apply the mobile agents for query retrieval process. In experiment three, the performance of mobile agent for query retrieval using the EHQR is better than the HQR approach. For example, the total time taken to retrieve 100 Kbytes of query results from 64 LP servers using the HQR approach is 55 s. However, the time taken to retrieve the same size of query results using the EHQR approach is 35.5 s. Approximately, 51.8% of time reduction has been achieved by using the EHQR approach compared with the HQR approach when applying the mobile agents for query retrieval process.

4.4 Conclusions

In this chapter, the performance evaluation of mobile agents for query retrieval based on the EHQR approach applied to the off-line MaSS has been proposed. The factors that affect the performance of the MaSS such as the number of mobile agents and the dispatching approach applied to each agents when using mobile agents to retrieve the query results have been investigated. Three experiments have been conducted to evaluate the performance of mobile agents in the MaSS. The first experiment is based on the comparisons between the on-line and off-line query retrieval approaches by mobile agents in the MaSS. As a result the off-line query retrieval approach performs better than the on-line query retrieval approach. The second experiment is based on the evaluation of the dispatching of mobile agents in the IQR, PQR, SHQR, and SQR approaches with 16 LP servers applied in the off-line MaSS. As a result the query retrieval approach using the PQR performs better than the others. In the third experiment, the comparisons of the query retrieval approaches applied to the off-line MaSS have been done using the HQR and EHQR approaches with 64 LP servers. As a result the EHQR performs better in the second experiment. Although three experiments have been conducted in order to evaluate the performance of query retrieval using the mobile agent technology, further issues related to mobile agents such as load balancing in the networks and data security are subjects to be explored in the future.

CHAPTER 5

CONCLUSION

5.1 Introduction

Mobile agents often have a task to collect data from several predefined sites. This should be done in an efficient way by minimizing the elapsed time. Usually these agents only know the list of sites but not the distances between them. This paper proposes a method to minimize a network routing time taken by the mobile agents to collect information from different sites using genetic algorithm (GA). The mobile agents repeat traveling over short routes and avoid longer ones. Mobile agents for query retrieval have used the GA to select the best routes that minimize the query retrieval time. The result shows that the proposed method provides good time minimization in retrieving the query results by the mobile agents based on different GA parameters.

5.2 Remarks

The factors that affect the performance of mobile agents in retrieving information from the Internet are the number of agents and the total of routing time taken by the participated agents to complete the assigned tasks. Fewer numbers of mobile agents used to execute the tasks will cause lower network traffic and consume less bandwidth, and the total time taken to retrieve the query results can be minimized. In this research, the performance of mobile agents for query retrieval is analyzed. Specifically, the performance of mobile agent in obtaining a query result from the remote hosts using extended hierarchical query retrieval (EHQR) approach is proposed. It is based on a hierarchical and a parallel dispatching of mobile agents to the remote servers in order to retrieve the query results. Experimental results show that the proposed approach reduces the number of mobile agents and also improve the total time taken to retrieve the query results compared with other approaches.

REFERENCES

[1] J. Sum, H. Shen, C-S. Leung, G. Young, Analysis on a mobile agent-based algorithm for network routing and management, IEEE Transactions on Parallel and Distributed Systems, 14 (3) (2003) 193-202.

[2] K. Kinoshita, T. Masuda, K. Kawano, H. Tanioka, T. Takine, K. Murakami, Network map architecture using inter-agent communication techniques for multiple QoS routing, IEICE Transactions on Communications, E85B (7) (2002) 1333-1341.

[3] W-S. E. Chen and C-W. R. Leng, A novel mobile agent search algorithm, in: Kurt Rothermel and Radu Popescu-Zeletin, editors, Mobile Agents (MA97), LNCS 1219, pp. 162-173, Springer Verlag, Berlin Heidelberg, 1997.

[4] Maes, P., Lashkari, Y. and Metral, M. (1997). Collaborative interface agents, Readings in Agents, edited by Michael N. Huhns & Munidar P. Singh, Morgan Kaufmann Publishers, Inc.

[5] Schoonderwoerd, R., Holland, O. and Bruten, J. (1997). Ant-like agents for load balancing in telecommunication networks, In Proceedings of the First Int. Conf. on Autonomous Agents, pages 209-216, ACM Press.

[6] Di Caro, G. and Dorigo, M. (1998). An adaptive multi-agent routing algorithm inspired by ants behavior, In Proceedings of PART98 - Fifth Annual Australasian Conference on Parallel and Real-Time Systems, Adelaide, Australia.

[7] Di Caro, G. and Dorigo, M. (1998). AntNet: Distributed Stigmergetic Control for Communication Networks, Journal of Artificial Intelligence Research, pp. 317-365.

[8] Selamat, A. and Omatu, S. (2002). Extended Hierarchical Mobile Agent for Query Retrieval, In Proceedings of Kansai-section Joint Convention of Institutes of Electrical Engineering, Kinki University, Osaka, Japan.

[9] Selamat, A, Omatu, S., Yanagimoto, H., Fujinaka, T. and Michifumi, Y. (2002). Effectiveness of Mobile Agent for Query Retrieval, IEEJ Transactions on Electronic and Information Systems, 122 (8): 1367-1373.

[10] Kashiwazaki, H. and Takai, H. (2002). Adaptive Network Routing by Using the Multiagent, Proc. of IASTED International Conference Networks, Parallel and Distributed Processing, and Applications.

[11] Chen, W.-S.E., Leng, C.-W.R. Lien Yao-Nan, (1997). A Novel Mobile Agent Search Algorithm, In the Sixth International Conference on Computer Communications and Networks (ICCCN '97).

[12] Sum, J., Shen, H., and Leung, C-S, Young, G., (2003). Analysis on a Mobile Agent-Based Algorithm for Network Routing and Management, 14:3, 2003, pp. 193-202

[13] M. Dikaiakos, M. Kyriakou and G. Samaras, Performance evaluation of mobile-agent middleware: a hierarchical approach, *Proceedings of the 5th IEEE International Conference on Mobile Agents*. In: J.P. Picco, Editor, *Lecture Notes in Computer Science Series* vol. 2240, Springer-Verlag, Berlin (2001), pp. 244–259.

[14] T. Kawamura, S. Joseph, A. Ohsuga and S. Honiden, Designing multi-agent systems based on pairwise agent interactions, *IEICE Transactions on Information Systems E* 84-D (2001) (8), pp. 968–980.

[15] S. Papastavrou, G. Samaras and E. Pitoura, Mobile agents for World Wide Web distributed database access, *IEEE Transaction on Knowledge Engineering* 12 (2000) (5), pp. 802–820

[16] L. Ismail, D. Hagimont, A performance evaluation of the mobile agent paradigm, in: Fourteenth Conference on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA), Denver, November 1999, pp. 306–313.

[17] W. Theilmann and K. Rothermel, Optimizing the dissemination of mobile agents for distributed information filtering, *IEEE Concurrency* 8 (2000) (2), pp. 53–61.

[18] R. Jha and S. Iyer, Performance evaluation of mobile agents for e-commerce applications, *Lecture Notes in Computer Science Series* vol. 2228, Springer-Verlag, Berlin (2001) pp. 331–340.

[19] P. Cogan, J. Gomoluch and M. Schroeder, Quantitative and qualitative comparison of distributed information processing using mobile agents realised in RMI and Voyager, *International Journal of Software Engineering and Knowledge Engineering (IJSEKE)* 11 (2001) (5), pp. 583–606.

[20] Voyager mobile agents website, Available from (http://www.objectspace.com/Voyager).

[21] Java website, Available from (http://www.java.sun.com).

[22] M.G. Rubinstein, O.C.M.B. Duarte and G. Pujolle, Evaluating the performance of a network management application based on mobile agents, *IFIP Networking 2002, Pisa, Italy*. In: E. Gregori, M. Conti, A. Campbell, G. Omidyar and M. Zukerman, Editors, *Lecture Notes in Computer Science* vol. 2345, Springer-Verlag, Berlin (2002), pp. 515–526.

[23] J.-J. Yoo, Y.-H. Suh and D.-I. LEE, Performance and scalability issues in mobile agent based workflow systems, *IEICE Transactions of Communications E* 84-B (2001) (10), pp. 2729–2739.

[24] Petri Nets. Available from (http://pdv.cs.tu-berlin.de/~azi/petri.html).

[25] I. Satoh, Itinerant agents for network management, *IEICE Transactions of Communications, E* 86-B (2001) (10), pp. 2865–2873.

[26] B. Schulze and E.R.M. Madeira, Migration transparency in agent systems, *IEICE Transactions of Communications, E* 83-B (2000) (5), pp. 942–950.

[27] R. Onishi, S. Yamaguchi, H. Morino, H. Aida and T. Saito, A multi-agent system for dynamic network routing, *IEICE Transactions of Communications, E* 84-B (2001) (10), pp. 2721–2728.

[28] A. Selamat, S. Omatu and H. Yanagimoto, Mobile agent search system for information retrieval from the Internet: MaSS, *Frontiers in Artificial Intelligence and Applications* 69 (2001) (2), pp. 914–918.

[29] Compaq Web Language: Available from (http://www.research.compaq.com/SRC/WebL/) (1997).

[30] D. Lange and M. Oshima, Programming and developing java mobile agents with Aglet mobile agent, Addison Wesley, Reading, MA (1998).

[31] S. Jones, Karen and P. Willet, Readings in Information Retrieval, Morgan Kaufmann, San Francisco, USA (1997).

[32] C. Ghezzi, G. Vigna, Mobile code paradigm and technologies: a case study, in: International Workshop on Mobile Agents, April, pp. 1997.

[33] Y. Aridor, M. Oshima, Infrastructure for mobile agents: requirements and design, in: Proceedings on International Workshop on Mobile Agents, 1998.

[34] W.R. Cockayne and M. Zyda, Mobile Agents, Manning Publications Co. (1998).

[35] D. Rus, R. Gray, D. Kotz, Autonomous and adaptive agents that gather information, in: AAAI'96 International Workshop on Intelligent Adaptive Agents, August 1996.

[36] S. Bandyopadhyay, K. Paul, Evaluating the performance of mobile agent-based message communication among mobile hosts in large ad hoc wireless network, in: Proceedings of the 2nd ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, August 1999.

[37] L. Miller, J. Yang, V. Honavar, J. Wong, Intelligent mobile agents for information retrieval and knowledge discovery from distributed data and knowledge sources, in: Proceedings of the IEEE Information Technology Conference, 1998.

[38] K.K. Sabnani, T.F. La Porta, T. Woo and R. Ramjee, Experiences with network-based user agents for mobile applications, *Mobile Networks and Applications* (1998).

[39] J.W. Baek, J.H. Yeo, G. Kim, H.Y. Yeom, Cost effective mobile agent planning for distributed information retrieval, in: Proceedings of the International Conference on Distributed Computing Systems (ICDCS), April 2001, pp. 65–72.

[40] AltaVista web search engine. Available from (http://www.altavista.com).

[41] Yahoo web search engine. Available from (http://www.yahoo.com).

[42] Excite web search engine. Available from (http://www.excite.com).

[43] A. Selamat, S. Omatu, H. Yanagimoto, T. Fujinaka and M. Yoshioka, Effectiveness of mobile agent for query retrieval, *IEEJ Transactions on Electrical and Information Systems* 122 (2002) (8), pp. 1367–1373.

[44] A. Selamat, S. Omatu, Extended hierarchical mobile agent for query retrieval, in: Proceedings of Kansai-section Joint Convention of Institutes of Electrical Engineering, Kinki University, Osaka, Japan, 9–10 November 2002.

[45] A. Selamat and S. Omatu, Web page feature selection and categorization using neural networks, *Information Sciences* 158 (2004), pp. 69–88.