Comparing Performance of Two Mobile Agent Platforms in Distributed Search

A. Mawlood-Yunis [*] A. Nayak [†] D. Nussbaum [‡] N. Santoro [§]

ABSTRACT

In this paper, we study the performance behavior of two mobile agent platforms (AGLET and TACOMA) in distributed search. The search problem is analyzed from two different standpoints: the single agent search and multiple agents search. Several experimental were carried out to measure the performance of mobile agents solution to distributed search problem under variable network size, network topology and number of agents. The results indicate that the two mobile agent platforms have similar behavior but their performance varies with underling implementation, and the multiple agents approach performs better than the single agent approach in large networks.

1 Introduction

A mobile agent is an entity which exists in a software environment. Mobile agents are particularly attractive approach for information retrieval in a distributed environment. The performance aspects of the mobile agents have been a subject of several studies [2, 3, 4, 7].

In this paper, we study performance behavior of mobile agents in AGLET [6] and TACOMA [5] platforms in a distributed search using single and multiple agents. The search has been conducted under variable network size, network topology and number of agents. Several experiments are carried out to measure the performance of mobile agents solution to distributed search problem.

2 The Prototype Architecture

The architecture consists of four main components (see Figure 1): *Blackboard*, *Whiteboard*, *Router*, and *Log*. The *Blackboard* is a multi-threaded HTTP server.

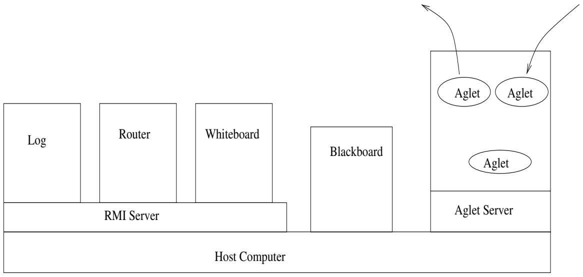

Figure 1: The Prototype Architecture

The purpose of the *Whiteboard* is to avoid multiple visit to the same node by different agents. The *Whiteboard* holds the state of the node specifying whether or not it has been visited. The implementation of the *Whiteboard* is based on the Remote Method Invocation (RMI) technology. The purpose of *Log* component is to terminate the flooding algorithm without using the *Blackboard* component. We also use it to hold information about the overhead time an agent spends in contacting *Blackboard*, *Whiteboard* and *Router*. Its implementation is similar to that of the *Whiteboard* and is based on the RMI technology. The *Router* component consists of a set of (RMI) objects, each associated with a node in the network. It acts as a global and local *Router* to enable agents to move around the network. The *Router* provides a way to recover from nodes failure as well.

3 Distributed Search

Since our objective is to compare the performance of TACOMA and AGLET, we use the same distributed search algorithm used and the results obtained by [8] for TACOMA. The search time of mo-

---

[*] School of Computer Science, Carleton University, Ottawa, Ontario, CANADA, K1S 5B6. Email: armyunis@connect.carleton.ca

[†] School of Information Technology and Engineering, University of Ottawa, 800 King Edward Avenue, Ottawa, Ontario, CANADA, K1N 6N5, Email: nayak@uottawa.ca

[‡] School of Computer Science, Carleton University, Ottawa, Ontario, CANADA, K1S 5B6, Email: nussbaum@scs.carleton.ca

[§] School of Computer Science, Carleton University, Ottawa, Ontario, CANADA, K1S 5B6, Email: santoro@scs.carleton.ca

bile agent we report in this paper does not include the time to construct the spanning tree, as we assume that the spanning tree has been constructed in preprocessing steps. Both *Breadth_First_Search* (BFS) and *Depth_First_Search* (DFS) traversal strategies are considered. We concentrate on the search algorithms instead of the search criteria, assuming that all nodes have to be visited. Agent interacts with the local resources at each node; it opens a file at each site it visits, and carries along the contents of the file. Once all nodes in the network were visited, the agent returns to the initiator node with the search result (file contents).

3.1 The Single Travelling Agent

At each host, the agent gets the list of the next hosts to visit by contacting the *Router*. That becomes the agent's *Itinerary*. The agent checks to see if a parent of the current host or initiator host are in the list. If the list contains the parent or initiator node, the agent removes it to avoid returning to the host it just came from or returning to the the initiator node before completing the search.

3.2 Flooding The Network With Multiple Agents

The agent starts from a node and duplicates itself as many identical agents as the number of children specified in the local number list; all these copies travel simultaneously to every child in the neighboring list.

4 Experimental Results

The strategy followed for testing different schemes consists of a set of test cases aimed at evaluating the impact of certain variables on the overall performance. The objective was to obtain sufficient information to compare the performance between the AGLET mobile agent and the TACOMA mobile agent. In the experiments, agents are injected from any node of the network with the purpose of performing a distributed search. The launching application or initiator measures the time in milliseconds before injecting the first agent and after the last agent has arrived. The difference between these two time measurements is the total execution time for the entire search.

All the experiments were carried out in the absence of failure. The computers were used simultaneously by other users, and no special care was taken to guarantee exclusive access to computer resources or network during these experiments.

4.1 Variable Network Size

We compared the search time of running single and multiple agents on binary trees of different sizes from 3 to 28 nodes. Each test was run thirty times to get the average search time. The following results were obtained:

1. The AGLET agent performed better than the TACOMA Agent. Searching with a single AGLET agent is almost ten times faster than searching with a single TACOMA agent. In multiple agent search, the AGLET agent performed almost seven times better than the TACOMA agent. This is due to the difference in their respective platforms.
2. For both the AGLET and TACOMA platforms, single agent performed better than multiple agents for small networks whereas multiple agents performed better than single agent in large networks. The only difference here is in the size of the network in which multiple agents outperformed the single agent. In the TACOMA platform, the single agent performed better than multiple agents for a network size up to eighteen nodes, while in the AGLET platform network size of only eight nodes were sufficient for multiple agents to outperform a single agent. This result suggests that we can run multiple AGLET agents on binary tree (i.e. take advantage of parallelism) on network with size as little as eight nodes, whereas we cannot do that with TACOMA agents. In order to take advantage of parallelism with TACOMA agents, we should have a network (i.e. binary tree) size of at least eighteen nodes.
3. The single agent search performance is better than the multiple agents search performance for networks of small size.

The above explanations lead us to conclude that for small networks, the advantages we gain from parallelism by using multiple agents is not enough to overcome the overhead associated with it. In large networks, the advantages of parallelism will overcome the above mentioned overhead, and as a result we get a better search performance with multiple agents than with single agent. Another aspect of the third point of the result is that in multiple agents search, the AGLET system performs better than TACOMA system in small networks. The performance comparison is shown in Fig. 2.

4.2 Variable Number of Children

In this test, we compared the results of running single and multiple agents of both TACOMA and AGLET systems on various trees with constant size. The trees are differ from each other in the number of children of each node. We started with a binary tree and then increased the number of neighbors for each node at each step by two until we ended up with a star structure.

The result shows that in the case of a single agent search, the number of neighbors did not have significant impact on search time in both systems. For both systems, multiple agents search performed better than single agent search when the number of neighbors was low. As the number of neighbors increased and the search became closer and closer to the sequential search, single agent performed better than multiple agent.

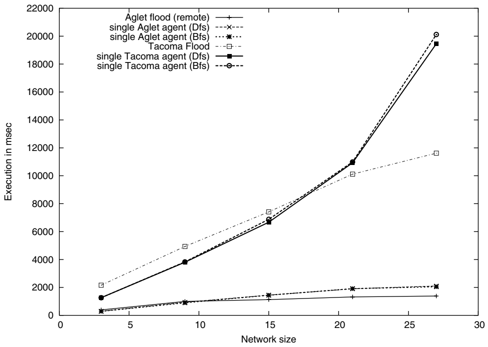

Figure 2: Comparison of AGLET and TACOMA Performance (Variable Network Size)

The only difference between AGLET and TACOMA agents here is at the turning point when single agent search starts to become more efficient than the multiple agents search. In the TACOMA system, multiple agents search performed better than single agent search only in the case of binary tree, whereas in AGLET system, multiple agents search performed better than the single agent search in a tree with each node having eight neighbors (k =8) and the best performance came in a tree with each node having four neighbors (k= 4).

Figure 3 shows the performance of AGLET and TACOMA when the network size is constant (N = 20) and the topology changes from a binary tree to a star (k =2 , 4, 6, . . . , 19).

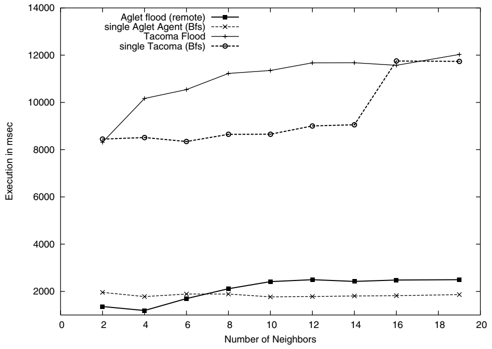

Figure 3: AGLET and TACOMA Performance (Variable Number of Neighbors)

4.3 Different Network Topologies

In this test, we compared the search times of running single and multiple AGLET and TACOMA agents in different network topologies of a fixed size. The network for this test consists of sixteen nodes. The topologies which been used for this test are: unidirectional ring, bidirectional ring, hypercube, and binary tree. Beside comparing the search times of AGLET and TACOMA, we also compared the search time of single and multiple agents for the AGLET system as well. We have the following observations:

Single Agent Traversal:

a. The best performance is seen in the bidirectional ring and the worst performance in the hypercube for both the AGLET and the TACOMA systems. Hence, the behavior of both systems is very much similar in different topologies when a single agent is used.
b. Even though we see the best and worst scenario, the performance in all topologies considered is almost the same for both systems.

Multiple Agent Traversal:

a. For deploying multiple agents, the best performance of the TACOMA agent is seen in the bidirectional ring, and the performance is similar in the case of a binary tree. On the other hand, the best performance of the AGLET agent is seen in the hypercube. From the first test, it is evident that the AGLET agents are more efficient than the TACOMA agents; hence, the AGLET agents are more capable of taking advantage of parallelism. The second test shows that the best performance of AGLET agents comes from a tree with four neighbors for each node or a hypercube.
b. The worst performance for the AGLET agent is seen in the unidirectional ring whereas the worst

performance for the TACOMA agent is seen in the hypercube.

Figure 4 shows the performance comparison of AGLET and TACOMA mobile agent in single agent search for different network topologies. Similarly, Figure 5 gives the result in the case of multiple agents.

5 Conclusions

In all the tests we conducted, single agent performed better than multiple agents in networks of small size, whereas in large networks, multiple agents outperformed single agent. The same result was seen in the TACOMA platform; the only difference between the AGLET and the TACOMA is in the size of the network in which multiple agents outperformed single agent. Comparing the results from both AGLET and TACOMA platforms, it appears that the overall behavior of a mobile agent is platform independent whereas its performance is platform dependent. For detailed prototype implementation other related results, the reader should refer to [1].

References

[1] A. Mawlood-Yunis, A. Nayak, D. Nussbaum and N. Santoro. On the Performance of Distributed Search by Mobile Agents. *Proc. 1st International Workshop on Mobility Aware Technologies & Applications*, 2004.

[2] R. S. Gray, D. Kotz, R. A. Peterson, J. Barton, D. Chacn, P. Gerken, M. Hofmann, J. Bradshaw, M. Breedy, R. Jeffers, and N. Suri. Mobile-Agent versus Client/Server Performance: Scalability in an Information-Retrieval Task. *Proc. 5th International Conference on Mobile Agents*, pp. 229-243 , 2002.

[3] R. Jain, F. Anjum and A. Umar. A Comparison of mobile agent and client-server paradigms for information retrieval tasks in virtual enterprises. *IEEE Academia/Industry Working Conference on Research Challenges*, pp. 209-215, 2000.

[4] R. Jha, and S. Iyer. Performance Evaluation of Mobile Agents for E-Commerce Application. *Proc. International Conference on High Performance Computing*, pp. 331-341, 2001.

[5] D. Johansen, R. V. Renesse and F. Schneider. An Introduction to the TACOMA Distributed System Version 1.0. Technical Report 95-23, Department of Computer Science, University of Tromso, Norway, 1995.

[6] D. B. Lange and M. Oshima. Programming and Deploying Java Mobile Agents with Aglets. Addison Wesley, Massachusetts, USA, 1998.

[7] M.G. Rubinstein, O. Carlos, M. B. Duarte and G. Pujolle. Scalability of a Network Management Application Based on Mobile Agents *Proc. 2nd International IFIP-TC6 Networking Conference*, pp. 515-526, 2002.

[8] E. Velazquez. Mobile Agents In Distributed Search: A Comparative Study. Master's Thesis, School of Computer Science, Carleton University, Ottawa, Canada, 2001.

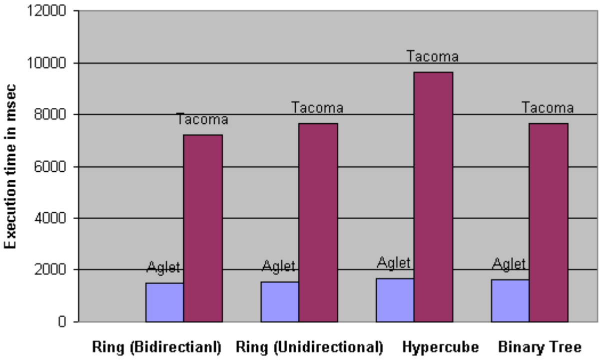

Figure 4: Performance of AGLET and TACOMA in Different Network Topologies (Single Agent Case)

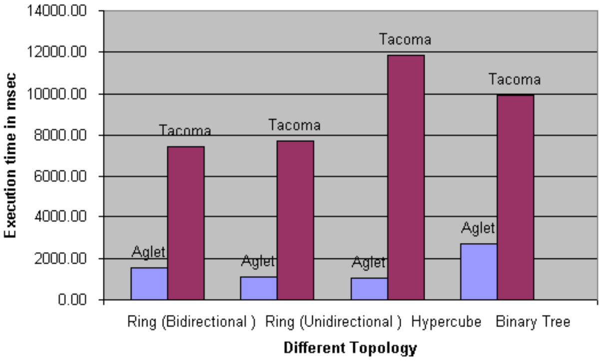

Figure 5: Performance of AGLET and TACOMA in Different Network Topologies (Flooding Case)