REVIEW PAPER
International Journal of Recent Trends in Engineering, Vol 2, No. 4, November 2009

MOBILE AGENT BASED DISTRIBUTED SYSTEM COMPUTING IN NETWORK

Prof. S. N. Gujar[1], Prof. G.R.Bamnote[2], Prof. R.S.Apare[2], Prof. M.A.Pund[2], Mr. S.R.Gupta[2]
[1]Smt. Kashibai Navale College of Engineering, Pune. India
Email: satish_gujar@yahoo.com
[2]Smt. Kashibai Navale College of Engineering, Pune.India
Email: ravi_apare@yahoo.co.in
[2]Prof. Ram Meghe Institute of Technology and Research, Badnera India
Email:{ pundmukesh@rediffmail.com , sunilguptacse@gmail.com , grbamnote@rediffmail.com}

Abstract
Mobile agents are emerging as a promising paradigm for the design and implementation of distributed applications. While mobile agents have generated considerable excitement in the research community, they have not translated into a significant number of real-world applications. One of the main reasons for this is lack of work that quantitatively evaluates the effectiveness of mobile agents versus traditional approaches. This paper contributes towards such an evaluation. This paper identifies the underlying mobility patterns of e-commerce applications and discusses possible client-server (CS) and mobile agent (MA) based implementation strategies for each of these patterns.

Keywords: Parallel Computing, Mobile Agent.

1. Introduction

Agent paradigm is a promising choice for network-centric applications, especially for Internet applications and services, because it is intrinsically communication and cooperation oriented [1, 2]. Agent concepts and mobile software agents have become a part of the system and service architecture of the next generation networks. It is where agent's mobility offers important advantages because of the network load reduction, increased asynchrony between the communicating entities and higher concurrency. Global end-to-end interactions, typical for a client-server paradigm, are replaced by local interactions in a server, visited by a mobile agent. Consequently, the need for long reliable connections is reduced; bandwidth requirements are lower and repeated interactions less frequent.

This paper elaborates performance issues of a mobile agent network [3]. It consists of a multi-agent system and its agents co-operate and communicate in a network that allows agent mobility. A mobile agent network is described as a queuing system where an agent represents an information unit to be served. Network flows of agents as well as the parameters defining agent hosting, execution and migration are introduced.

2. Current Research

Parallel and distributed computing on a cluster of workstations is being increasingly applied to a variety of large size computational problems. Several software systems have been developed that make distributed computing available to an application programmer. Examples of run-time environments are PVM [6], pPVM [6], Linda [7], P4 [8], Express [9], they all support distributed computing in varying degrees of generality; however, none of them is web based, lack collaborative features, and are mostly suitable for running SPMD programs.

The second category of available environments which addresses large distributed heterogeneous applications are: FIDO and MIDAS [10, 11].However, both these systems are either hardwired to a specific problem or are too restrictive. The other major limitation is that they lack a collaborative environment which allows dierent members in a group to interact with the application at various stages of its execution.

There are Web based computing environments based on client server [12, 13] framework.

3. Advantages of Mobile Agent

The following are the primary advantages of Mobile Agents:

▪ Facilitate high quality, high performance, and economical mobile applications. Applications employing Mobile Agents transparently use the network to accomplish their tasks, while taking full advantage of resources local to the many machines in the network. They process data at the data source, rather than fetching it remotely, allowing higher performance operation. They use the full spectrum of services available at each point in the network, such as GUIs at the user and database interface on servers. They make best use of the network as they travel.

▪ Enable use of portable, low-cost, personal communications devices. Network support, including security, is contained in a lightweight server which manages the movement of agents in the network. Coupled with the sophisticated, self-contained programming model afforded by agents, which permits a small footprint to be achieved on user devices, without sacrificing functionality for the application.

▪ Permit secure Intranet-style communications on public networks. Security is an integral part of the Mobile Agent framework, and it provides for secure communications even over public networks. Agents carry user credentials with them as they travel, and these credentials are authenticated at every execution point in the network. Agents and their data are fully encrypted as they traverse the network. All this occurs with no programmer intervention.

▪ Efficiently and economically use low bandwidth, high latency, even on error prone communications channels. The agent network employs a store and forward mechanism to transfer agents between nodes. This is well suited to the problematic nature of many communications channels, especially in the mobile arena. Queuing and persistent checkpoints enhance this further, to the point that agents can use such channels with no degradation in reliability or response. Because the agent data processing takes place at the point of source, the network has no effect on the agent as it executes.

4. Quantitative Evaluation of MA versus CS

The same setup for quantitative evaluation of e-commerce application for mobile agent and traditional CS approaches can be used. The CS implementation consists of a server that sends a catalog on request and a multithreaded client that requests a catalog from one or more servers. The client and the server can be implemented using Java sockets. Voyager framework was used for MA implementation of these evaluation experiments. The results are as follows:

· **Effect of catalog size on turnaround time:** The processing delay at the server was kept constant and catalog sizes of 100KB, 200KB, 500KB and 1MB were used. This was done for different scenarios of product discovery from 1 to 26 shops.

· **Effect of server processing delay on turnaround time:** The catalog size was kept constant at 1MB and the server delay was varied from 20ms to 1000ms. The user turnaround time was measured for different scenarios of product discovery from 1 to 26 shops.

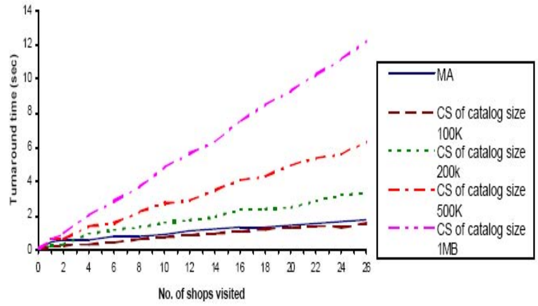

Fig.1. Effect of catalog size on turnaround time for sequential MA & sequential CS

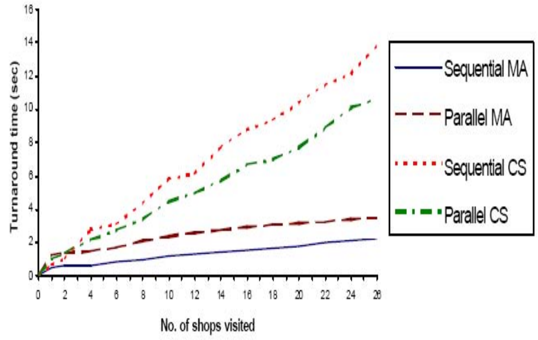

Fig.2. Turnaround time for a processing delay of 20 ms (catalog size of 1 MB)

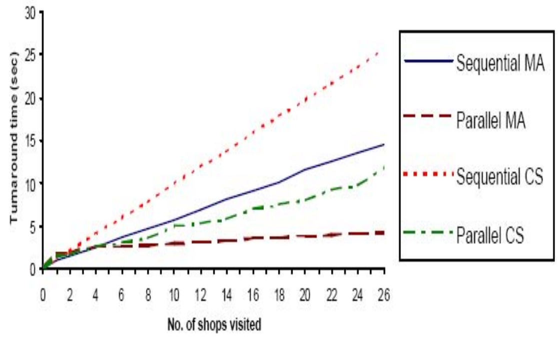

Fig.3. Turnaround time for a processing delay of 500 ms (catalog size of 1 MB)

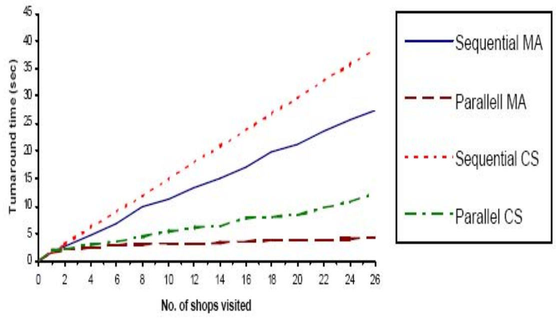

Fig.4. Turnaround time for a processing delay of 1000 ms (catalog size of 1 MB)

5. Observations

The results of our MA versus CS evaluation experiments are shown in the graphs of Fig1to 4. Some key observations are:

- The performance of MA remains the same for different catalog sizes while the performance of CS degrades with increase in catalog size (Fig. 1).
- CS implementations perform better than MA implementations for catalog sizes less than 100 KB (Fig. 1).
- MA performs better than CS when the catalog size is greater than 200KB and number of shops to visit is greater than or equal to 3 (Fig. 1).
- MA performs better than all other strategies, for small processing delays (20 ms) and large (1MB) catalog size (Fig. 2).
- Parallel implementations perform better than sequential implementations when the number of shops to visit is greater than or equal to 6 and the processing delay is greater than or equal to 500ms (Fig. 3).
- Parallel MA performs better than parallel CS for higher processing delays (1000ms) and large (1MB) catalog size (Fig. 4).
- *Performance crossover points* i.e. parameter values for which MA starts performing better than CS implementation can be found for a given set of ecommerce application parameters.

6. Conclusions

This paper suggest that CS implementations are suitable for applications where a "small" amount of information (less than 100 KB) is retrieved from a "few" remote servers (less than 4), having "low" processing delays (less than 20ms). However, most real-world e-commerce applications require large amount of information to be retrieved and significant processing at the server. MAs scale effectively as the size of data to be processed and the number of servers the data is obtained from increase. Scalability being one of the needs of net-centric computing, we find that MAs are an appropriate technology for implementing e-commerce applications. Parallel implementations are effective when processing delay (greater than 1000ms) contributes significantly to the turnaround time. It also identifies *performance crossover points* for different implementation strategies; this could be used to switch between implementations for performance critical applications. The *hybrid model* employing all the four identified implementation strategies would result in high performance gain for large-scale distributed applications.

7. References

[1].Y.Aridov and D.Lang, Agent design patterns: element of agent application design, Proc. Autonomous Agents '98, pp108-115. ACM, '98.
[2]. Antonio Carzaniga, Gian Pietro Picco and Giaovanni Vigna, Designing Distributed Applications with Mobile Code Paradigms, Proc. 19th International Conference on Software Engineering (ICSE '97) , pp.22-32, ACM , '97.
[3]. David Kotz & Robert S Gray. "Mobile code: The future of the Internet", Proc. Autonomous Agents, May 1998.
[4]. Alfonso Fuggetta, Gian Pietro Picco and Giovanni Vigna, Understanding Code Mobility, IEEE Transactions on Software Engg, vol. 24(5),
[5]. Carlo Ghezzi & Giovanni Vigna, Mobile code paradigms and technologies: A case study, Proc. First International Workshop on Mobile Agents, Berlin, Germany, vol. 1219, April 1997.
[6] A. Beguelin, J. Dongara, G. A. Geist, R. Manchek, and V. Sunderam. A Users' Guide to Parallel.Virtual Machine. Technical report, Oak Ridge National Laboratory, TR ORNL/TM-11826, Oak Ridge, TN 37831, June 1991.
[7] N. Carrieo & D Gelernter. Linda in context. Communication (ACM, Vol 32 No. 4, April 89).
[8] Dimple Bhatia, Vanco Burzevski, Maja Camuseva, Georey Fox, Wojtek Furmanski and Girish Premchandran, WebFlow - a visual programming paradigm for Web/Java based coarse grain distributed computing, http://www.npac.syr.edu/projects/webbasedhpcc/index.html
[9] A. Kolawa, The Express programming Environment. Workshop on Heterogeneous Network based Concurrent Computing, Tallahassee, October 1991.
[10] Thomas M. Eidson, Robert P. Weston, FIDO - Framework for Interdisciplinary Design Optimization, http://hpccp-www.larc.nasa.gov/do/homepage.html
[11]. D.Lange and M. Oshima, Mobile Agents with Java: The Aglet API, Proc. Web Processing: Applications and Techniques of Web Agents, Baltzer Science Publishers, 1998.
[12] Haresh S. Bhatt, V. H. Patel and A. K. Aggarwal, Web enabled client-server model for development environment of distributed image processing, http://cisr.anu.edu.au/sam/java/jav.mpi.prototype.html.
[13] Albert D. Alexandrov, Maximilian Ibel, Klaus E. Schauser, and Chris J. Scheiman, SuperWeb: Towards a Global Web-Based Parallel Computing Infrastructure, fberto ibel,schauser,chrissg@cs.ucsb.edu