Resource Management in Wireless Ad-hoc Networks using Mobile Agents

Monideepa Roy, Aparajita Chandra, Bipin Behari Nandi, Sujoy Mistry, Nandini Mukherjee

Abstract

Resource brokering among mobile devices connected through an ad-hoc network poses several challenges, mainly because of their constraints of resources and mobility. This paper proposes an environment where mobile agents can be used for migrating a job to a service provider node for execution, when a client device is unable to process it on its own due to inadequate resources. Here the service providers form an ad hoc network and register their resource parameters with a leader node. Each service provider also maintains an updated and consolidated resource information list of all the other devices, among themselves. When a thin client searches for a suitable match to run the job, the job is allocated to the first most suitable node on the list. The selected node starts execution of the job, but if it cannot complete it for any reason then it migrates the job to the next most suitable node on the network.

Keywords : multi-hop, mobile agents, resource management, mobile Grid, ad hoc network

I. Introduction

Mobile computing [1] has evolved from its initial functionality of merely providing access to information, communications and services everywhere, anytime and by any available means into being a much more versatile technology. With time, the expectations and requirements from mobile devices have risen from just the using of routine services and information to also being able to perform compute intensive tasks.

At the same time, Grid technologies have also evolved keeping in mind the heavy computational and storage requirements of the clients who need to process their compute or storage intensive jobs.

A natural consequence of proliferation of the above two technologies has been the creation of the paradigm of "mobile Grid computing" [2], which joins together the concept of the Grid with the pervasiveness of mobile computing. The major emphasis of all the research efforts in this area has been put forward on two aspects. The first is how to access Grids from personal mobile devices, while on the move. The second is how to incorporate a number of mobile devices into a service provider Grid and offer services to other client mobile devices that do not individually have the resources to perform the tasks assigned to them. In this paper we propose a mobile Grid environment based on ad hoc networks where the service providers and the clients are connected wirelessly. The following two scenarios can be considered as examples of the application of the Grid environment.

(i) In the first case we suppose that a person wants to execute a particular job on his/her device but does not have the adequate resources to do so. In that case the device can search for and register with any available wireless ad hoc network which is constituted of more resource-capable devices and submit the job for execution.

Department Of Computer Science & Engineering
Jadavpur University, Kolkata -32.
monideepa_roy@in.com, aparajita.chandra@yahoo.co.in
bipin.ju@gmail.com , sujoy.mtech@gmail.com ,
nandinmukherjee@yahoo.co.uk

[In a traditional Grid environment, often a user with insufficient resources can also avail of additional resources by registering as a Grid user.]

(ii) In the second case, we consider a group of mobile device users who have come together and want to share the resources among themselves for the execution of a job. They form a wireless ad hoc network and complete the task.

[This idea is similar to the traditional Grid computing concept of resource sharing among heterogeneous devices.]

In the above two situations, we assume that the mobile devices have limited mobility, that is once they are included in a Grid environment, they remain within the environment for a specific period of time (we refer to the time period as the lease time) and when they move out, the tasks assigned to them are migrated elsewhere. Mobility of the devices is not within the scope of this paper, so rather we focus on the resource management within the environment. Furthermore the resource management strategy presented in this paper is a distributed one and does not depend on a single central node.

The rest of the paper is organized as follows: Section II defines mobile agents, Section III describes the environment and the experimental setup, Section IV discusses the results and its analysis and finally the future work and conclusion are presented in Section V

2. Ad Hoc Networks And Mobile Agents

A wireless ad hoc network [3] [4] [5] [16] (Figure-1) is a self configuring network of devices connected together by wireless links. Each device in the network is free to move independently in any direction, and may therefore change its links to other devices frequently.

The nodes in an ad hoc network have several constraints like network coverage area, limited computational resources and battery life [6]. Replicating the Grid environment to work with such resource constrained devices poses challenges that are inherent to the highly variational wireless networks. Therefore, achieving the same levels of robustness and QoS that exists in a wired network is hardly feasible in a wireless environment.

So, a job which is too large or compute-intensive to be executed on a particular node, can be sent to another higher configuration node for execution or can be migrated over multiple nodes and executed in parts. The selection of the nodes is usually according to the resource or cost constraints imposed by the user. Some of the parameters that may be used for checking the suitability of the service provider node are resource accessibility, system workload, network performance etc. One of the prevalent ways in which a job is migrated from one node to another is through the use of 'mobile agents'.

Mobile agents have the ability to move from one system to another in order to process their tasks locally on that system and return to their point of origin [7] [8] [9] [10]. Mobile agents are defined as autonomous programs with the ability of moving from host to host and acting on behalf of its creators towards the completion of a given task. A mobile agent is not bound to the system where it begins execution. It has the *unique ability to transport itself* from one system in a network to another. The ability to travel permits a mobile agent to move to a destination agent system that contains an object with which the agent wants to interact. Moreover, the agent *may utilize the object services* of the destination agent system.

In this paper we propose to migrate jobs from one device to another in a wireless ad hoc network and get them executed on the devices which possess adequate resources for execution of the jobs.

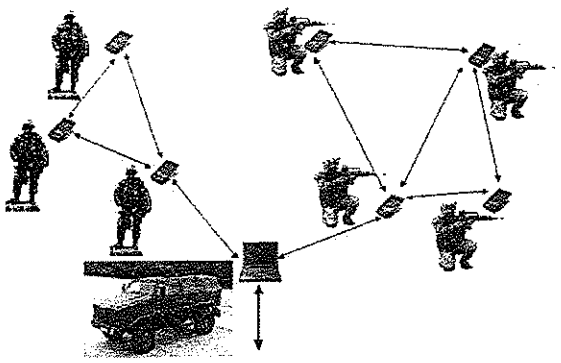

Figure 1: A wireless ad hoc network

3. Overview Of The Environment

In our proposed mobile Grid environment, the ad hoc wireless network consists of several mobile nodes, which are designated with the following different roles:

Leader node

A leader node [11] [12] [17] is a node, which waits for new mobile nodes which come within its range and registers the new nodes in the network. The primary role of a leader node is to balance the resource consumption among all the registered nodes in the network and prolong the lifetime of a wireless ad hoc network.

The node which first arrives in a site, where a mobile Grid environment will be created, is usually selected as a leader node.

Service provide node

It is a node which is capable of and is willing to provide a service (may be computational service).

Client node

A client is a thin node which wants to execute a particular job but does not have the necessary resources to do so.

The activities within the environment can be broadly classified as: (i) discovery and registration, (ii) job submission, (iii) resource brokering, and finally (iv) job execution with or without migration. In this section, we give an overview of each of the above activities:

A. Discovery & Registration:

When the service provider node enters within the range of a leader node, it sends a request for registration to the leader node [13] [14]. The leader node then dynamically allocates an IP to the service provider. The service provider node then sends its resource information (such as *number of processors, and memory size*) to the leader node in an XML format (Figure 2). The consolidated resource information of all these service provider nodes is stored in the form of an XML file which is named as AllDetails.xml. (Figure 3)

After each registration, the leader node updates the XML file and sends a copy of this updated file to all the registered nodes at that moment in the network. Although storing, updating and matching of the XML files incur overheads of time etc. as the file sizes increase, the average number of devices registered on a MANET is generally not expected to be more than ten to fifteen at a time. So the overheads are not very substantial here.

Thus, after the initial communication between the leader nodes and the registered mobile nodes, all the nodes have information about the availability of resources in all the other mobile nodes within range.

```
<xml>
    <Hostname>
        <Noofprocessors>2</Noofprocessors>
        <TotalMemory>5177344</TotalMemory>
        <FreeMemory>4981440</FreeMemory>
        <IPAddress>169.254.56.03</IPAddress>
    </Hostname>
</xml>
```

Figure 2. The XML file sent to the lookup server

A client enters into an ad hoc network and finds a leader node and registers in a similar manner.

```
<xml>
    <Host-name>
        <No-of-processors>1</No-of-processors>
        <Total-Memory>5177344</Total-Memory>
        <FreeMemory>4969912</FreeMemory>
        <UsageMemory>207432</UsageMemory>
        <IPAddress>169.254.17.250</IPAddress>
    </Host-name>
    <Host-name>
        <No-of-processors>2</No-of-processors>
        <Total-Memory>5177344</Total-Memory>
        <FreeMemory>4981440</FreeMemory>
        <UsageMemory>195904</UsageMemory>
        <IPAddress>169.254.211.142</IPAddress>
    </Host-name>
</xml>
```

Figure 3 : AllDetails.xml file sent to the registered node

B. Job and resource requirement submission:

When a client needs to execute a job which is beyond its resource capabilities, it enters its job requirements through a GUI and for which an XML file is created on the client machine itself. The client now runs a search for the IP of the machine with the best device match from the AllDetails.xml file and allocates the job to the selected device for execution. The client subsequently sends a notification to the leader node informing it of the IP of the machine where it has sent the job for execution. An SLA is established between the client and the resource provider and leader node receives a copy of the SLA. In this way, the leader node keeps track of all the job scheduling decisions, though the actual decision is taken in a

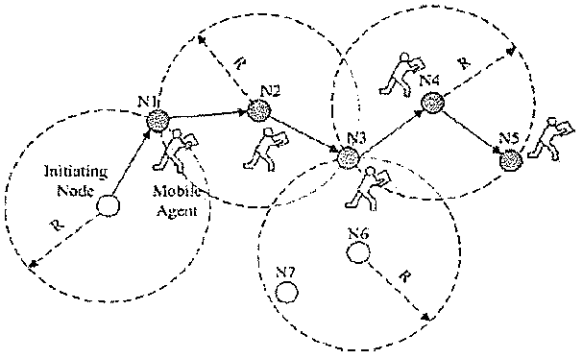

Figure 4 : Job migration through mobile agents

C. Resource brokering:

As described above, the resource brokering in the proposed system is distributed. The selection process of the best suitable node is such that it is expected to meet the resource or cost constraints as imposed by the user, i.e. the clients' requirements are matched with the AllDetails.xml file for selection of prospective nodes which can host the job. The list of the authorized devices and the resources that are available in the network and the information about the job loads allocated to them are continuously updated. This information can be later used for performance monitoring and load balancing in the network.

D. Job Migration and Execution:

After finding a suitable node for execution of a job, the job is migrated to that node using mobile agents. In order to demonstrate the process of the execution of a job, *three* cases are considered with four nodes in each, labeled as node 1, 2 3 and 4 respectively:

i. Single hop client server machines

A client node submits a job to the leader node in that network which is capable of executing the job. Then the job is executed by the leader node itself. Finally it returns the result back to the client node (Figure 5).

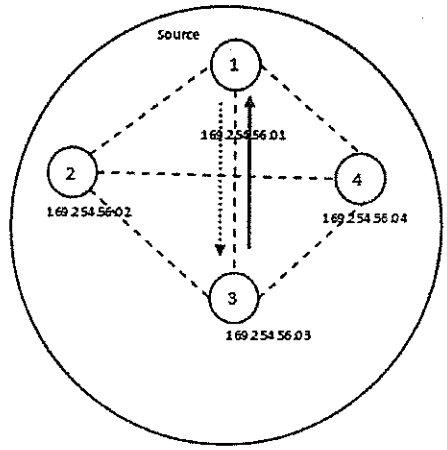

Figure 5 : Single hop job migration

ii. Multi-hop with job migration and entire execution at final node

In this case, a client checks the AllDetails.xml file to find a suitable node for scheduling the job and submits the job to that node. However, after migration of the job to the selected node, it may so happen that the node is unable to execute the job due to some problems, like the arrival of a higher priority job or unexpected resource failure. Thus, even before beginning its execution, the job needs to be migrated to another suitable node. The job may be migrated through multiple intermediate nodes but is executed only at the final node which is eventually found capable of executing it (Figure 6).

iii. Multi-hop with migration with sequential and partial execution on multiple nodes

In a similar case, a client migrates the job to a suitable node. But in this case the job execution progresses on the first node for some time, and then if this node has to suspend the execution for any reason, then the partially executed job is sent by this node to the next node (selected from the AllDetails.xml file stored on the current node) for execution. The job is check-pointed and migrated using mobile agents and the execution starts on the new node from the point where the execution was suspended. Thus, a job may be actually executed on several nodes and finally the node which completes its execution sends the response back to the client. In Figure 7, the job is partially executed through nodes 4 and 3 and the response is sent back to node 1.

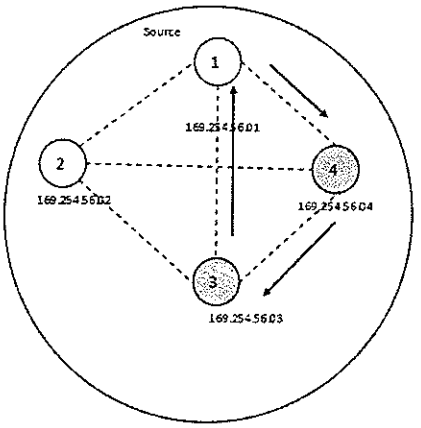

Figure 6: Multi-hop job migration over multiple nodes and entire execution at final node

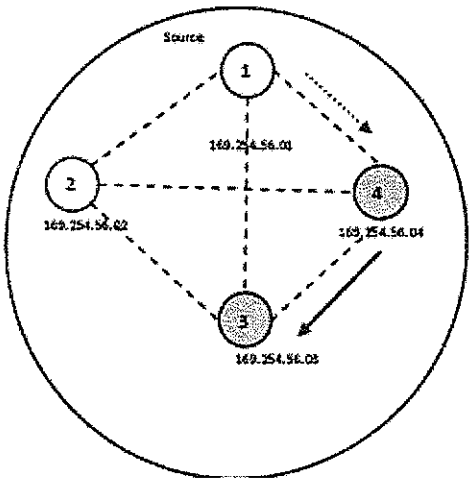

Figure7 : Multi-hop job migration and sequential and partial execution on multiple nodes

4 Result Analysis

We have carried out some preliminary experiments in order to demonstrate and predict the performances of the above strategies and techniques [15]. For our experiment purpose, 4 desktop PCs and a laptop have been used. All the devices were wirelessly enabled using D-Link cards and form an ad hoc wireless network.

A bubble sort algorithm has been implemented and tested for the three cases discussed in Section 4. In the first case,

there is only one service provider to which the client directly submits the job for execution and gets back the result. The experiment was repeated several times and a graph representing the turnaround times for each execution of the job is shown in Figure 8.

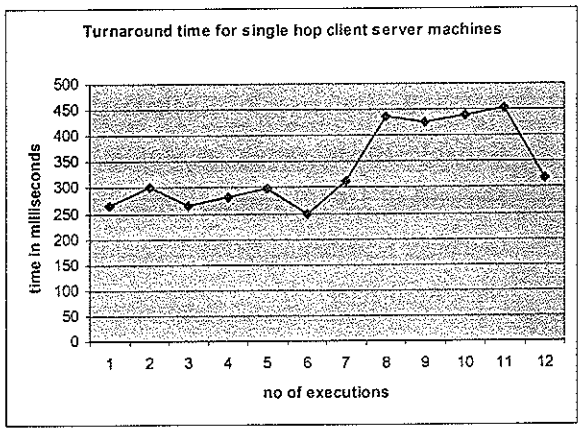

Figure 8 : Turnaround time for Single hop

The next two figures present the experimental results in the second and third cases. In each case, the bubble sort program has been executed repeatedly and the turnaround times were measured for each execution.

In the second case, the job has been migrated over nodes 1, 2 and 3 and finally executed on the last node. Figure 9 shows the graph representing the turnaround times in the situation where the job is migrated over multiple nodes but the execution occurs only at the final node.

The third case has been tested for migration over nodes 2, 3 and 4. Figure 10 shows the graph of the turnaround times in the situation where the job is migrated over multiple nodes, with partial executions at each node. At present here the job is equally divided among the nodes, so that each node sequentially executes the job in equal proportion. In reality, this will be decided dynamically and will depend on the actual condition of the nodes.

The results show that the turnaround times are more or less stable over multiple executions of the same job, with all other parameters remaining the same and the minor differences in the time maybe attributed to delays owing to network congestions.

But when the number of nodes that are traversed by the mobile agents increases, the turnaround time also increases, which is primarily because of migration overheads. However the positive change is that the predominantly flat slopes of the lines indicate that the timings for different node values vary by a more or less constant factor and can therefore be kept within limits. This factor seems to be an acceptable one, since the primary need for a job migration is assumed to be the fact that the initial node is incapable of executing the job at all. Moreover, the migration overheads may become negligible if the experiments are carried out with larger jobs, i.e. if the execution time of the job is significant compared to our implementation of the bubble sort with a small input array.

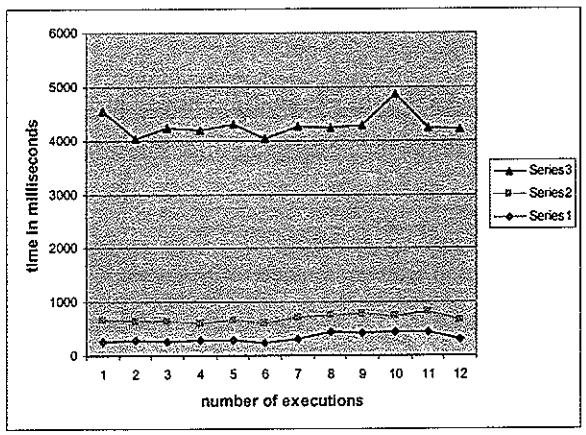

Figure 9 : A comparative graph showing the turnaround time with migration over 1, 2 and 3 nodes, with entire execution at final node

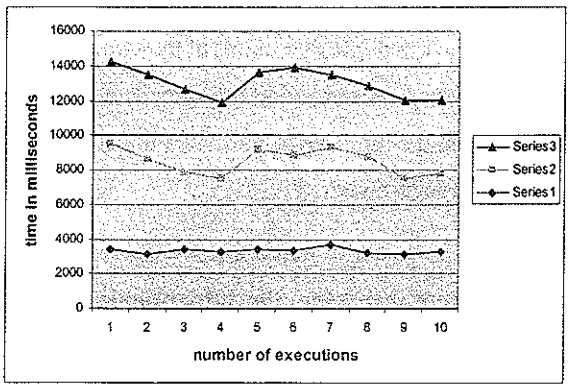

Figure 10 : A comparative graph showing the turnaround time for migration and partial execution over 2, 3 and 4 nodes

5. Future Work

We are in the process of extending this work to resource matching within multiple ad hoc networks and handling the issues of mobility of the nodes. Moreover, when a service provider registers to provide a service, we introduce an SLA (Service Level Agreement) for it. We are also working on the structure of the SLA. More experiments with a larger environments and bigger programs are required. Also currently we are performing sequential execution of a job on multiple nodes. Parallel execution of the jobs on multiple nodes in a similar environment is another interesting area of research.

References

[1] Frank Adelstein, Sandeep K.S. Gupta, Golden G. Richard III, Loren Schwiebert, " Fundamentals of Mobile and Pervasive Computing", Tata McGraw-Hill

[2] Zhi WANG, Qi Chen, Chuanshan GAO "Implementing Grid Computing over Mobile Ad-Hoc Networks based on Mobile Agent" Department of Computer Science and Engineering, Fudan University Proceedings of the Fifth International Conference on Grid and Cooperative Computing Workshops (GCCW'06) 0-7695-2695-0/06 $20.00 © 2006

[3] Magnus Froigh, Per Johansson and Peter Larsson "Wireless ad hoc networking-The art of networking without a network", Ericsson Review No. 4, 2007.

[4] S.V.G Bavithiraja, R. Radhakrishnan "A Reliable Broadcasting in Mobile Ad Hoc Networks" IJCNS International Journal of Computer Science and Network Security, VOL.9 No.4, April 2009

[5] Sinead Cummins, J.P.O Grady & Fergus O'Reilly "Managing Wireless Ad-Hoc Networks, IP Addressing and Service Delivery" Adaptive Wireless System Group, Department of Electronic Engineering, Cork Institute of Technology

[6] David A. Maltz "Resource Management In Multi-hop Ad Hoc Networks", School of Computer Science, Carnegie Mellon University, Pittsburg, PA 15213 November 21, 1999 CMU-CS-00-150.

[7] Danny B. Lange General Magic Inc.420 North Mary AvenueSunnyvale, CA 94086 U.S.A., Mitsuru Oshima IBM Tokyo Research Laboratory 1623-14 Shimotsuruma, Yamato-shi Kanagawa-ken 242, Japan "Mobile Agents with Java: The Aglet API".

[8] Nikos Migas, William J.Buchanan, and Kevin A. McArtney, "Mobile Agents for Routing, Topology Discovery, and Automatic Network Reconfiguration in Ad-hoc Networks", School of Computing Napter University,EH105DT, Scotland

[9] R. Tobias Meier, Jurgen Dunkel, Yosiaki Kakuda and Tomoyuki Ohta "Mobile Agents for Service

Discovery in Ad Hoc Networks" 1550-445X/08, 2008 IEEE DOI 10.1109/AINA 2008.78

[10] www.javaworld.com/javaworld.com/javaworld/jw-06-2002/jini

[11] Noman Mohammed, Hadi Otrok, LIngyu Wang, Morrad Debbabi and Prabir Bhattacharya, "Mechanism Design-Based Secure Leader Election Model for Intrusion Detection in MANET", Computer Security Laboratory, Concordia Institute for Information System Engineering, Concordia University, Montreal, Quebec, Canada

[12] Orhan Dagdeviren and Kayhan Erciyes "A Hierarchical Leader Election Protocal for Mobile Ad Hoc Networks" Izmir Institute of Technology Computer Eng. Dept., Ege University International Computer Institute, Turkey

[13] Gang Ding and Bharat Bhargav, "Peer-to-peer File Sharing over Mobile Ad hoc Networks" Department of Computer Sciences, Purdue University West Lafayette, IN 47907, USA

[14] L. Cheng: "Service Advertisement and Discovery in Mobile Ad Hoc Networks" Workshop on Ad Hoc Communications and Collaboration in Ubiquitous Computing Environments, New Orleans, USA, 2002.

[15] Madhavi W. Subbarao "Ad Hoc Networking Critical Features and Performance Metrics" Wireless Communications Technology Group, NIST October 7, 1999

[16] www.home networkhelp.com/ wireless-network.html

[17] Sukhor Abd Razak, Normalia Samian, Mohd. Aizaini Ma'arof, S. M Furnell, N. L Clarke, P.J Brooke, "A Friend Mechanism for Mobile Ad Hoc Networks", Journal of Information Assurance and Security 4(2009) 440448

Author's Biography

Professor Nandini Mukherjee received the PhD degree in Computer Science from University of Manchester, Manchester, United Kingdom in 1999, and the Master in Computer Science & Engineering degree from Jadavpur University, Kolkata, India in 1991, and the Bachelor of Engineering in Computer Science & Technology from Bengal Engineering College, Sibpur, India in 1987. She also worked as Post-doctoral Research Associate at the Department of Computing Science, University of Newcastle, United Kingdom. Since 1992 she has been a faculty of the Department of Computer Science and Engineering, Jadavpur University, Kolkata, India. She is also the director of School of Mobile Computing and Communication, Jadavpur University. Her research interests are in the area of High Performance Parallel Computing, Grid Computing and Mobile Computing. She is working as secretary of Computer Chapter, IEEE Calcutta Section, and also working as Secretary of Institute for Open Technology and Applications. She acted as a member of the organizing committee and program committee of many international conferences.

Monideepa Roy did her M.Sc. in Mathematics & Computer Science from IIT Kharagpur in 1996. She is a Research Fellow and currently doing her Ph.D. from Jadavpur University. Her areas of interest are wireless networks, mobile computing and cloud computing.

Aparajita Chandra has done her BE in Computer Science and Engineering from West Bengal University of Technology (Birbhum Institute of Engineering and

Technology) in 2005, and her Masters in Computer Science and Engineering from Jadavpur University in 2009. She has been working as an Assistant System Engineer at Tata Consultancy Services Ltd. since September 2009.

Bipin Behari Nandi has done his BE in Computer Science and Engineering from Jadavpur University in 2009. He has been working in an EDA company, Atrenta India Pvt Ltd as Software Engineer since June 2009. His areas of interest are Java, C++, Mobile Agent, Wireless Networking, Hardware Languages, Design Automation and Verification.

Sujoy Mistry received his B.Sc. in Computer Science from Calcutta University in 2005, and M.Sc. in Computer & Information Science from Calcutta University in 2007. He did his M.Tech. in Computer Science and Engineering from Calcutta University in 2009. He is currently working as a Junior Research Scholar (intern) in the School for Mobile Computing and Communication (SMCC), Jadavpur University.