CHANDIGARH GROUP OF COLLEGES
Campus : Jhanjeri, Mohali

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

A Novel Technique for Handling Mobility in Distributed System

Harpreet Kaur[1]

Abstract

Distributed systems are built up on top of existing networking and operating systems software. A distributed system comprises a collection of autonomous computers, linked through a computer network. The whole task is divided on number of resources. The user is not aware that the jobs are executed by multiple computers subsist in remote locations. Distributed computing system is a network of both wired and mobile nodes. Though mobile agents offer much more flexibility, but it has some additional costs and issues such as reliability, security, and fault tolerance which required to be addressed for successful adaptability of mobile agent technology for developing real life applications "Mobility is the one of the problem in the Distributed system which is discussed in this paper". Mobility problem occurs when node moves in between performing the task in Distributed Computing System. This problem leads to the fault occurrence problem and degrades network performance. The existing algorithm is the master and slave architecture through which the task are assigned by the master node to slave nodes and slave nodes execute task as assigned by the master nodes. In such type of architecture, the divide and rule technique is followed, and after executing the task slave nodes will revert back to master nodes. The major problem in this architecture is of fault, if one slave node moves during task execution then the task allocated by master node will not get completed and fault occurred. In this paper a Nobel technique is developed to handle mobility of nodes in distributed systems.

Keywords: distributed computing system, fault tolerance, task reallocation, and mobility.

1. M.Tech Student, Department of Computer Science and Engineering, SGGSWU, Fatehgarh Sahib, Punjab, India

Introduction

Distributed computing is a network of many computers, each accomplishing a portion of an overall task, to achieve a computational result much more quickly than with a single computer. Distributed Computing System is a system which is consists of one or more computers and associated software with common storage. A distributed system connected by local networks and physically connected with each others. Distributed Computing System is heterogeneous in nature. So different type of hardware and software are required are required to build the distributed system. Distributed system is better than centralized system in the following manners [11].

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

- **Scalability:** By adding more machines as needed the system can easily be expanded.

- **Redundancy**: Several machines can provide the same services, so if one is unavailable, work does not stop. Additionally, because many smaller machines can be used, this redundancy does not need to be prohibitively expensive.

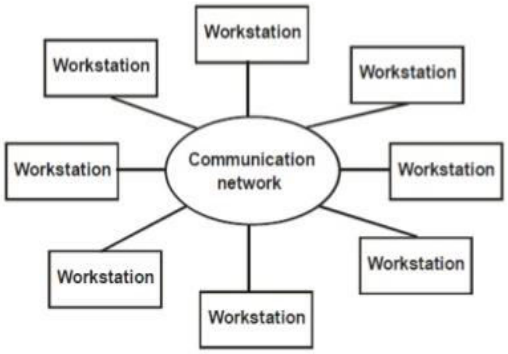


Fig. 1: Distributed Computing System

A distributed system as a collection of independent computers that appear to the users of the system as single computer. There are two essential points in this definition. The first is the use of the word independent. This means that, architecturally, the machines are capable of operating independently. The second point is that the software enables this set of connected machines to appear as a single computer to the users of the system. This is known as the single system image and is a major goal in designing.

1) Architecture of Distributed Computing

The architecture of distributed computing is as follows:

a. Fabric: This consists of all the distributed resources, owned by different individuals and organizations, shared on the system. This includes workstations, resource management systems, storage systems, specialized devices, etc.

b. Resource and Connectivity Protocols: This contains core communication and authentication protocols that provide secure mechanisms for verifying the identity of users and resources and allow data to be shared between resources

c. Collective Services: This contains Application Programming Interfaces (APIs) and services that implement interactions across collections of resources. This includes directory and brokering services

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

for resource allocation and discovery, monitoring and diagnostic services, application scheduling and execution and more.

d. User Applications: This contains programming tools and user applications that depend on mobile resources and services during their execution.

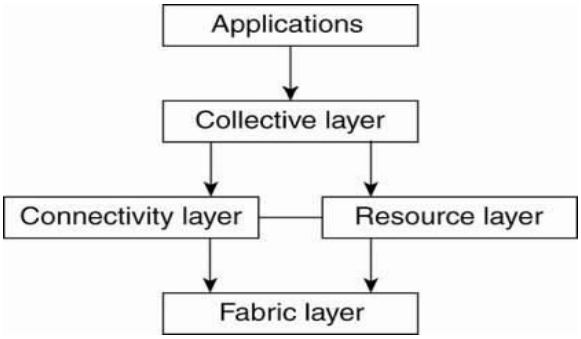


Fig. 2 : Architecture of Distributed Computing

2) Mobile Computing System:

Mobile computing is a class of distributed systems, both wired and mobile agents, that autonomously engage in sharing of resources and exchange of services to provide higher performance. As the resources on the host are limited with power and memory constraints, it seems unnecessary for extending certain features of grid to the mobile host. But in future, when the systems become all-pervading, the constraints of power and memory would seem trivial when compared to the advantages offered by mobile grid. Besides, with continuous development in device technology, the constraints of power and memory are diminishing and mobile devices are having better specifications when compared to their predecessors. Mobile devices can make use of the available resources in the grid to perform any task [12].

Though mobile agents offer much more flexibility, but it has some additional costs and issues such as security, reliability, and fault tolerance which need to be addressed for successful adaptability of mobile agent technology for developing real life applications as these agents move in between performing the task.

3) Fault Tolerance in Distributed System:

A distributed computing system must be fault tolerant. It should be able to continue its functioning in the presence of faults. Fault tolerance describes a component or computer system that is designed in a way that in the event if a component fails, a backup procedure or component can immediately take its place with no loss of service. A highly fault-tolerant system might continue at the same level of performance even though one or more components have failed. It is the capability of a system to respond gracefully to an unexpected software and hardware failure. Fault tolerance is a major concern

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

to guarantee reliability and availability of critical services as well as application execution. Fault tolerance is very good method that provides us greater reliability by applying group of hardware like processors resources and communication host [5].
Faults can be classified into one of three categories:

a. Transient faults: These faults occur once and then vanish. For example, a network message doesn't reach its destination but does when the message is redeliver.

b. Intermittent faults: Intermittent faults are characterized by a fault occurring, then disappearing again, then reoccurring, then disappearing. These can be the most annoying of component faults. A loose connection is an example fault.

c. Permanent faults: This type of failure is persistent: it continues to exist unless the faulty component is repaired or replaced. Some examples are disk head crashes, software bugs, and burnt-out power supplies.
Fault Tolerance can be achieved with the help of two ways. These ways are as follow:

- Recovery
- Redundancy

A good fault- tolerant system design requires a careful study of failures causes of failures and system responses to failures. Such learning should be approved out in aspect before the design start and have to remain part of the design process [16].

Planning to keep away from failures is most important. A designer must examine the situation and decide the failures that must be tolerated to achieve the preferred level of dependability. To optimize fault tolerance, it is important to calculate approximately actual failure rate for each possible failure.

4) Problem Statement

Distributed computing is a network of many independent computers, each performs a portion of an overall task, to achieve a computational result much more quickly than with a single computer. In distributed system, the master node divides the overall task into multiple sub- nods and these sub-nodes complete the task according to the given instruction of the master node. As distributed systems gain complexity owing to increasing user needs, monitoring and adaptations are necessary to keep them fit and running. Mobile agent technology has become a new paradigm for distributed systems. Mobile entities connected by a wireless link, without any fixed support in the network so the network disconnection is very frequent between the mobile nodes. Due to which network's topology will change suddenly. Due to above reasons chances of errors in the mobile distributed network is very high.
When mobile nodes move from one location to another location during task execution, node failure problem occurred. To handle this fault which occurs due to mobility of node, reallocation of the task to the most reliable candidate node is needs to be done.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

5) Literature Survey

G Attiyaet.al (2006) [1]paper addresses the problem of task allocation in heterogeneous distributed systems with the goal of maximizing the system reliability.

P K Yadav et.al (2011) [2] explained that in Distributed computing systems (DCSs), task allocation strategy is an essential phase to minimize the system cost.

Shan Zhang et.al (2009) [6] mentioned criteria based on the combination of the description function of metadata, the distributed computation function of web service, the centralized storage ability of database, the GIS server component function of ArcGIS Server and the spatial database engine function of ArcSDE, the open management of shared dataset's catalogue and the problem of interoperating distributed and heterogeneous data were solved.

Vinod Kumar et.al (2012) [7] solve the problem of maximizing reliability of heterogeneous distributed computing system where random node can fail permanently.

Zhongkui Li et.al (2013) [8] considers the distributed control problem for heterogeneous multi-agent systems with matching uncertainties and a leader whose control input might be nonzero and not available to any follower.

Rajwinder Singh et.al (2013) [13] propose a novel parallel check pointing algorithm antecedence graph approach for achieving fault tolerance in mobile agent systems. By recording the dependency relation among mobile agents in antecedence graphs and check pointing those to stable storage during the normal computation message transmission, the proposed algorithm can reduce the time latency for a global check pointing procedure significantly. Furthermore, it only forces the minimum number of MAs to take their checkpoints and minimizes the number of blocked mobile agents during identifying, which improves the system performance compared with previous graph based approaches.

Tome Dimovski et.al (2011) [10]proposed a Connection Fault- Tolerant Model for mobile environment which considers two communication scenarios.

Asma Insaf Djebbar et.al (2009) [12] presented the mobile Ad hoc networks are distributed environments characterized by a high mobility and limited battery resources. In these networks, mobiles nodes are subject to many errors. In this paper, we present our approach of modeling by groups for faults tolerance based in MAS, which predicts a problem and provide decisions in relation to critical nodes. Their work contributes to the resolution of two points. First, they propose an algorithm for modeling by groups in wireless network Ad hoc. Secondly, they study the fault tolerance by prediction of disconnection and partition in network.

6) Proposed Methodology

Distributed computing system consist of various autonomous computers, each computer perform some part of whole task, to get the result more quickly as compared to single computer. Mobility is the one of the problem in the Distributed system. Though mobile agents offer much more flexibility, but it has

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

some additional costs and issues such as reliability, security, and fault tolerance which required to be addressed for successful adaptability of mobile agent technology for developing real life applications. In distributed system the master node divide the overall task into slave nodes but when the mobile node moves during task execution then the reallocation of the task needs to be done.

A node failure problem occurs due to mobility of the node. In the existing algorithm when the task arrives to the master node then master node divides the task and allocate to the candidate nodes. The candidate nodes are chosen on the basis of failure rate and minimum execution time. Here Master node set threshold value which includes two parameters one is failure rate and other is maximum execution time. The sub nodes which have equal to and less failure rate and execution time are selected as candidate nodes by the master node.

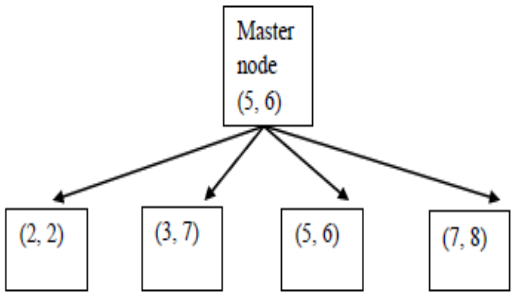


Fig. 3 : Selection of candidate node

In fig 3 master node has set threshold value of Maximum failure rate and maximum execution time is (5, 6).After that failure rate and execution time of each candidate nodes is entered.

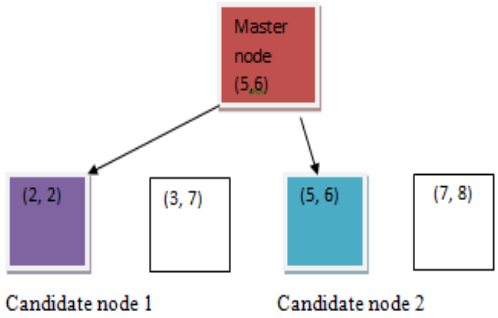


Fig. 4: Candidate node are selected according to failure rate and execution time

In fig 4 the node which has value equal to and less this threshold value is selected as a candidate node. N1 has smaller value than threshold value so it will be a candidate node. N2 has one parameter less and other is high so it will not be chosen as a candidate node. N3 has value equal to the threshold than it will be selected as a candidate node 2. Again N4 has a greater value than threshold value than will not be selected as candidate node. After the selection candidate node will start perform their tasks. We will also enter number of task in this scenario. Suppose during execution of task one moves from its

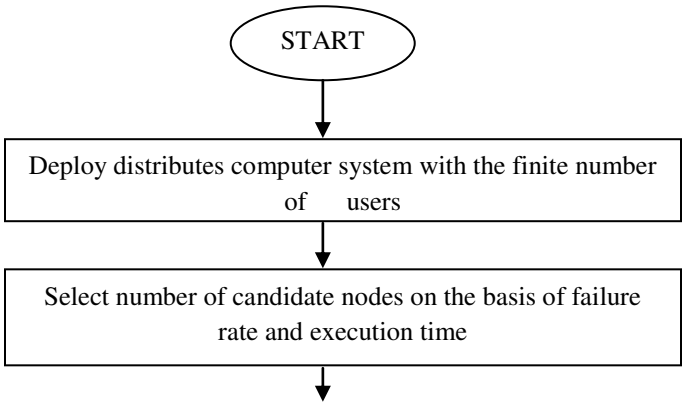
International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

location than failure occurs at that point. To overcome this problem a novel technique has been proposed which overcome the problem of failure due to mobility of the node.
In the proposed algorithm, we have added a new parameter in the present algorithm that is master node time and number of tasks that arrives at master node. Master node time is the result time to join the end users. It is for node collaboration by using the execution time , failure rate ,master node time and the number of tasks the efficiency of each candidate node is calculated. When the candidate node moves in between performing the task leaving the task uncompleted then efficiency of the remaining candidate node is calculated and the node with highest efficiency is assigned the uncompleted task. then after completing the task candidate node send the result back tom the master node.

7) Purposed algorithm

1. Start
2. Define Distributed system with defined number of nodes.
3. Enter number of task to be executed.
4. Enter the master node time.
5. Enter maximum execution time and failure rate of the master node
6. Enter the failure rate and execution time of each candidate node.
7. Select best candidate node for the allocation of the task.
8. To select best candidate node comparison is made between master node and each candidate node.
9. Candidate nodes which have less failure rate and execution time than master node will be selected as best nodes for task execution.
10. Selected candidate nodes starts, execution of the task.
11. If candidate node move during the execution of task than fault is occurred and the task is note completed.
12. Calculate efficiency of each candidate node.
13. Reassign the task to candidate node with highest efficiency.
14. Recovery node starts executing the task and return output.

8) FLOW CHART



International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

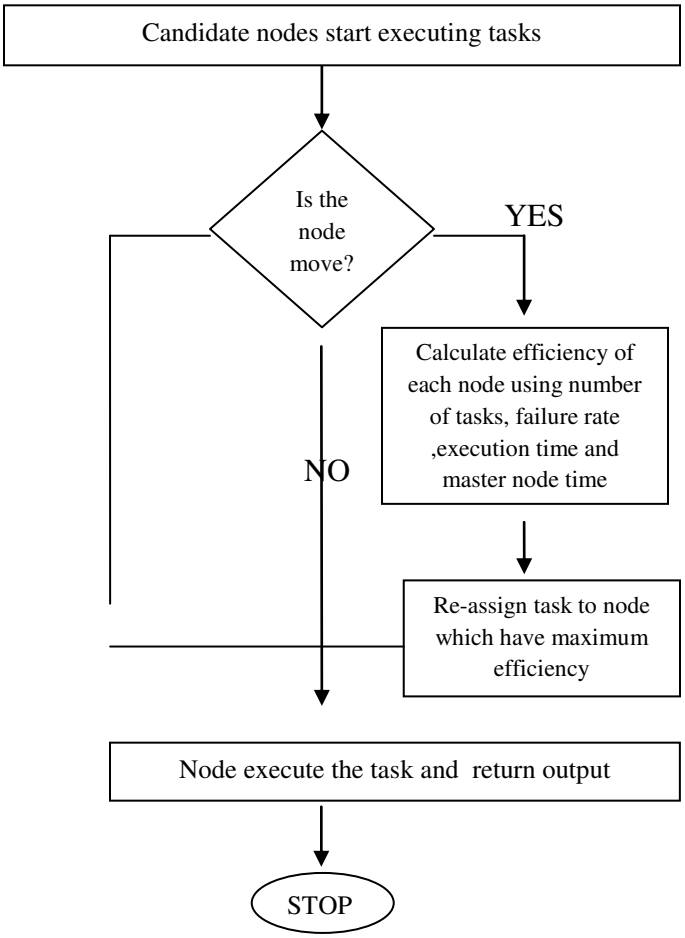


Fig. 5 : Flowchart

9) Experimental results

In this section, results of the existing technique that is Distributed task allocation and after enhancing the Distributed task allocation by using Efficiency based algorithm have been shown. The comparison has been made between the existing and proposed technique on the basis of the parameters: processing time, resource consumption, energy Consumption.

a. Processing Time Graph

Table 1: Comparison table of Processing time

Number of tasks	Processing time (seconds)	
	Existing algorithm	Proposed algorithm

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

6	32	22
10	48	40
15	55	45

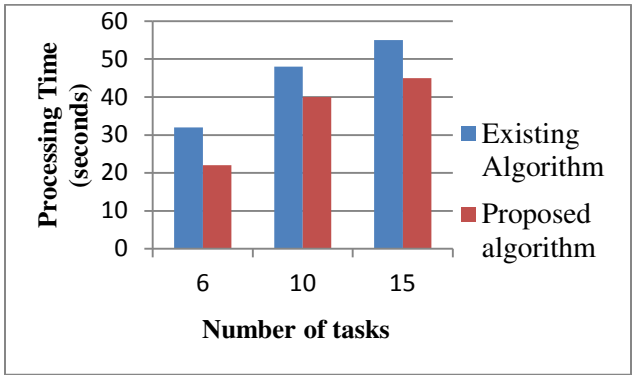


Fig .6: Processing Time Graph

As illustrated in figure 6, the processing time of existing scenario in which fault is occurred due to node mobility is shown and processing time of proposed algorithm is shown in which fault is recovered. In the existing scenario fault is occurred and master node is keep on waiting for the task completion. The proposed algorithm will reassign the task to most appropriate node and it will complete the task due to which processing time of proposed algorithm is less as compared to existing scenario. Each time when user enters different number of tasks, processing time of proposed as well as existing algorithm gets changed, but for each input, the processing time of the proposed algorithm will be better than the existing algorithm.

b. Resource Consumption Graph

Table 2:Comparison table of Resource Consumption

Number of tasks	Resource consumption (bytes)	
	Existing algorithm	Proposed algorithm
6	16	11
10	25	20
15	31	28

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

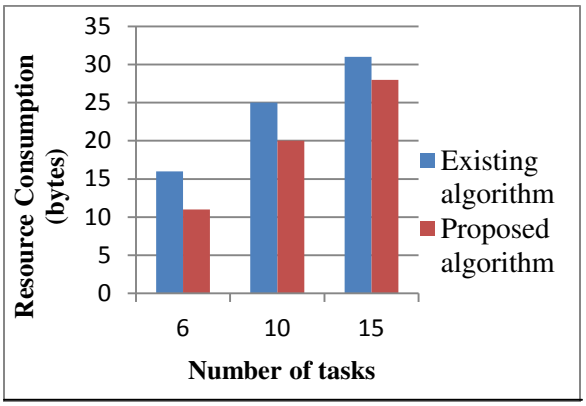


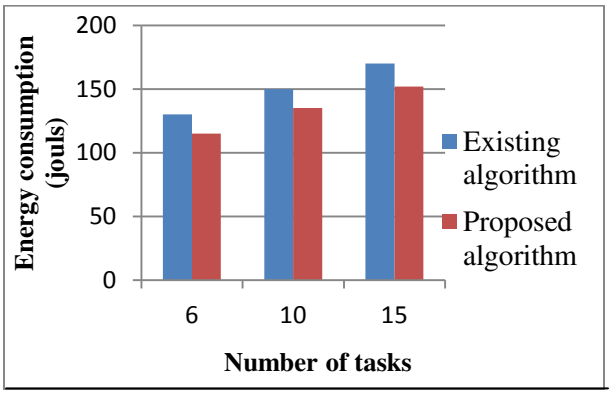
Fig. 7: Resource Consumption Graph

As shown in figure 7, the resource comparison is shown in terms of buffer size. In the existing and proposed algorithm resources are assigned to candidate nodes for task execution. Due to the fault occurrence in the existing algorithm the task will not be completed so the resource consumption is more as compared to the proposed algorithm where the fault is removed by reallocating the task. Each time when user enters different number of tasks, resource consumption of proposed as well as existing algorithm gets changed.

c. Energy Consumption Graph

Table 3: Comparison table of Energy Consumption

Number of tasks	Energy consumption (joules)	
	Existing algorithm	Proposed algorithm
6	130	115
10	150	135
15	170	152



International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

Fig.8: Energy Consumption Graph

As shown in figure 8, the energy comparison is made between the existing and proposed scenario. It is been analyzed that when fault is occurred in the network, master node is keep on waiting for the task to complete due to which energy consumption of the network is high. When task is re-assigned to other candidate node, tasks get completed on time due to which, energy consumption of the network is less. Due to the increases in processing time the energy consumption also increases and vice versa.

10) Conclusion

In this paper, a novel technique has been proposed which overcome the fault of mobility in the network. When the node moves in between performing the task leaving the task unfinished ,then that task is reallocated to other candidate node by using the efficiency technique. This approach will enhance the network performance, reduce execution time and reduce battery consumption. The proposed algorithm is based on the number of task, failure rate, execution time and the time taken by the master node for fault recovery.

11) Future Work

In future, enhancement in the proposed algorithm will be made to handle certain security attacks in mobile distributed systems; these attacks are denial-of-service attacks which reduces the networks reliability and efficiency

References

[1] G. Attiya, "Y Haman, "Task allocation for maximizing reliability of distributed systems: A simulated annealing approach", Journal of Parallel and Distributed Computing, vol. 66, no. 10, pp. 1259-1266,October 2006.

[2]P. K.Yadav, M P Singh, K Sharma. Article, "An Optimal Task Allocation Model for System Cost Analysis in Heterogeneous Distributed Computing Systems", Interntional Journal of Computer Applications, vol.28, no. 4, pp.30-37, August 2011.

[3]K.Sheel, A. Kumar, "Some Issues,Challenges and Problem of Distributed Software System", International Journal of Computer Science And Technologies, vol.5, no. 4, pp. 4922-4925, June 2014.

[4]V.singhal,D.Dahiya, "Distributed task allocation in dynamicmulti-agent system", International Conference on Computing Communication and Automation, vol.4, no. 2, pp. 643-2648, May2015.

[5] J. colouris, Dollimore,T.Kindberg, "Distributed Systems : Concepts and Design", Addison-Wesley,Pearson Education 3rd Edition 2001.

[6] S. Zhang, J. W. ping, "Construction of Distributed and Heterogeneous Data Sharing Platform", International Conference on Web Information Systems and Mining, vol. 60, no. 2, pp. 696- 700, November 2009.

International journal Of Business Management
ISSN NO. 2349-3402
Available at www.ijbm.co.in
VOL. 3 Issue 1, 2017

[7] V. K.Yadav, M. P.Yadav, D. K.Yadav, "Reliable Task Allocation in Heterogeneous Distributed System with Random Node Failure: Load Sharing Approach", IEEE vol.62, no. 2, pp. 187-192, September 2012.

[8] Z.Li, Z.Duan, "Distributed Tracking Control of Multi-Agent Systems with Heterogeneous Uncertainties" , 10th IEEE International Conference on Control and Automation (ICCA) Hangzhou, vol. 65, no.4, pp. 1956-1961, June2013.

[9] J.Ahn, "Lightweight Fault-tolerance Mechanism for Distributed Mobile Agent-based Monitoring",IEEE, vol.62, no. 4, pp.1-5, March 2008.

[10] T.Dimovski, P.Mitrevski, "Connection Fault-Tolerant Model for Distributed Transaction Processing in Mobile Computing Environment" , ITI 2011 33rd Int. Conf. on Information Technology Interfaces,Cavtat, Croatia, vol. 62, no. 3, pp. 145-150, June 2011.

[11] R. Singh, M. Dave, "Using Host Criticalities for Fault Tolerance in Mobile Agent Systems, 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, vol.60, no. 4, pp. 67-72, December 2012.

[12] A. I.Djebbar, G. Belalem, "Modeling by groups for faults tolerance based on multi agent systems", International Conference on Machine and Web Intelligence IEEE, vol.62, no. 3, pp.35-40, October 2010.

[13] R. Singh, M. Dave, "Antecedence Graph Approach to Checkpointing for Fault Tolerance in Mobile Agent Systems", Transactions on computers IEEE, vol.62, no. 3, pp. 247-258,Feburary 2013.

[14] A. S. Tanenbaum,R. V.Renesse, "Distributed Operating Systems", vol. 17, no. 4, pp. 419-470, December 2006.

[15] K.Govil, "A Smart Algorithm for Dynamic Task Allocation forDistributed Processing Environment" International Journal of Computer Applications, vol.28, no. 2, pp. 13-19, August 2011.

[16] A.K. Verma, M.T. Tamhankar, "Reliability-based optimal task-allocation in distributed-database management systems", Transactions on reliability IEEE, vol. 46, no. 4, pp 452-459, December 1997.