International Journal of Electrical & Electronics Research (IJEER)
Volume 3 issue 2, Pages 9-14, June 2015, ISSN: 2347-470X

Fault Tolerance Techniques in Heterogeneous Mobile Distributed Computing System- A Review

Madhav Gupta[1], Dr. Ruchi Singla[2]
[1]Student (M.Tech Scholar, ECE), Chandigarh Engineering College, Mohali, India
[2]Professor and Head, Department of ECE, Chandigarh Engineering College, Mohali, India
madhavgupta890@gmail.com[1] , ruchisingla@yahoo.com[2]

Abstract- In Distributed computing system, Fault tolerance is an important issue because if the system fails then whole execution of a tasks stop. Fault tolerance is that asset of a system which provides the service to perform well still in case of any faults. A task applied on the real time distributed system must be feasible and reliable. The real time distributed systems for instance grid networks, robotics, air traffic control systems, etc. exceedingly depends on time. A single error in real time distributed system can cause a whole system failure, if not detected accurately and recovered at the proper time. Fault-tolerance is the key method which is often used to provide continue reliability in these systems. By applying extra hardware like processors, resource, communication links hardware fault tolerance can be achieved. A fault perhaps will occur for numerous reasons in distributed computing system such as failure of network, hardware or software failure etc.

This paper defines various terminologies like failure, fault, faulty environment, fault tolerance, candidate node, redundancy, etc and explains fundamental concepts linked to fault tolerance in distributed systems. There are a lot of issues in distributed Computing system such as Emergent resource sharing, transparency, dependability, Complex mappings, concurrency, Fault tolerance etc. In this paper we focussed on the different fault tolerant approaches and fault tolerant terminologies used in distributed computing environment.

Keywords- Distributed System, Reliability, Fault tolerance, Faulty environment**.**

1. INTRODUCTION

Distributed computing system is a system with multiple software components attached to multiple computers, but appears as a single computing system. Distributed computing system makes a network of various computers, each accomplish a portion of a taken as a whole task, to accomplish desired output faster than a single system. The resources or components used in the distributed computing system are attached to each other by a local network and may or may not be physically linked to each other. A distributed system allows workstations to match their activities and to share resources of the system so that users observe the system as a single built-in computing facility. Distributed computing systems are much better than centralized computing systems in following manner:

Scalability: Distributed systems can easily be expanded by adding up more machines as required.

Redundancy: In distributed system failure of one machine or component does not stop the work because several machines give the same services. Moreover, many smaller machines can be used this redundancy does not requires to be costly

1.1 Parallel computing

Distributed computing Systems offer a platform for executing parallel applications. In parallel applications calculations are carried out at once. Parallel computing is a method of multiple uses of computer resources to solve computational problems extremely simple.

Parallel computing is fast, time saving and is the future of computing. There are numerous algorithms of parallel computing. Single Instruction Single Data (SISD), Single Instruction Multiple Data (SIMD), Multiple Instruction Single Data (MISD), Multiple Instruction Multiple Data (MIMD) are its taxonomy.

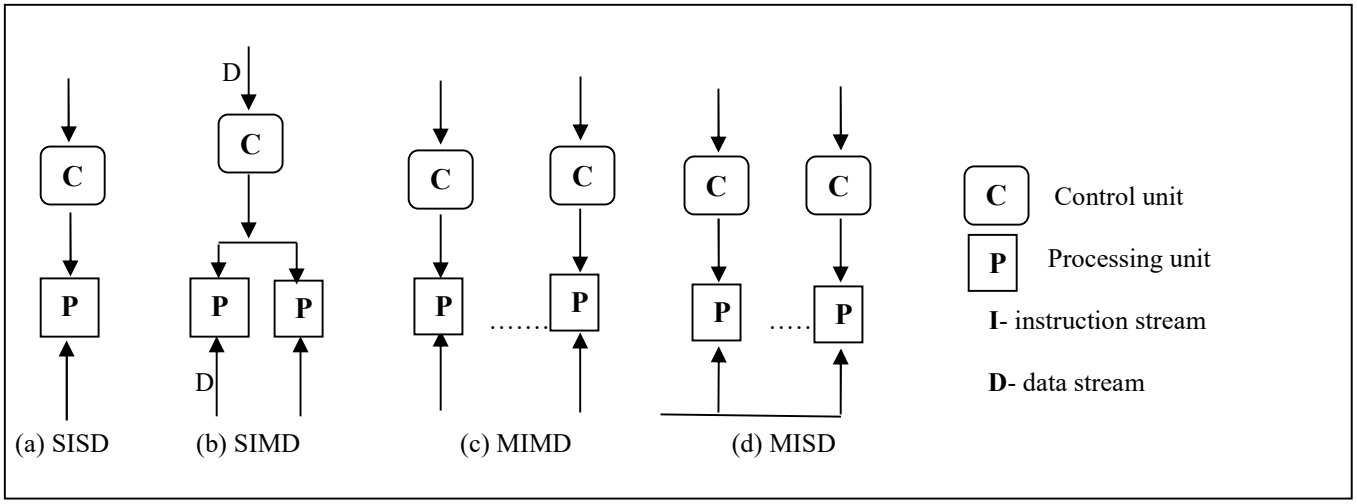

Fig 1.1 Relation to parallel multiprocessor/multicomputer systems.

1.2 Distributed Computing System

The Distributed Computing System (DCS) is the set of heterogeneous and geographically dispersed computing elements that cooperatively execute the applications. To process the application, DCS firstly, divide them into set of tasks. Then it concurrently processes to these tasks on different processors of DCS. In literature, the problem of tasks assignment is called task allocation problem. Various tasks may have different hardware and software requirements. So they will perform their expected functionality when their requirements are met. If the allocation of tasks hasn't done carefully, processors of the system use more time to transfer the tasks before performing useful computations. This problem in literature is called Thrashing. Different types of hardware and software are required to build distributed system.

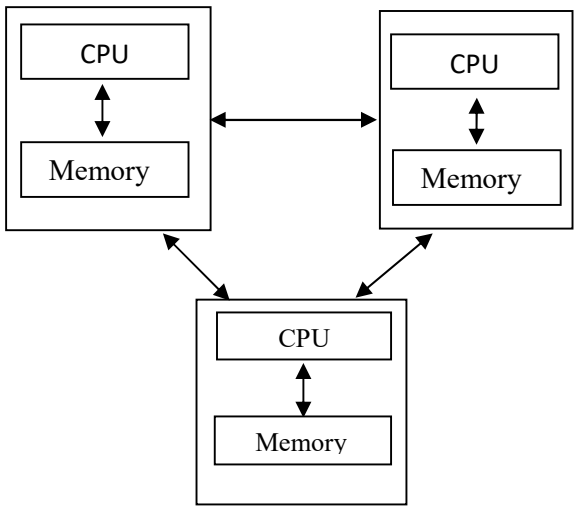

Fig 1.2 Distributed computing system.

A computer program that runs on distributed system is known distributed program. The process of writing such types of languages is called distributed programming. Grid computing and Cluster computing are types of distributed computing systems.

2. FAULT AND FAULTY ENVIRONMENT

In this section we will discuss some basic concepts about fault and faulty environment and finally fault tolerance. Fault can be termed as "error" at the lowest level of abstraction. In order to give a better efficiency and performance, a system must detect the fault and perform efficiently even in the case of fault. The different types of faults that can occur in distributed computing system are as: Network Fault, physical fault, Media faults, Process faults and Processor faults.

A Fault that can take place in a network as a result of partition of network, Loss of packets, corruption of packets, failure of link, etc. is known as *Network Fault.* The Fault which can take place in hardware such as fault in CPUs, Fault in memory, Fault in storage, etc. is known as *physical fault.* Fault occurs as a result of media head crashes are known as *Media faults.* Fault occurs in processor because of operating system crashes, etc. is known as *Processor faults*. Fault occurs by reason of shortage of resource, software bugs, etc. is known as *Process faults.*

Fault appears with regard to time are as follows:

Permanent: These faults occur by accidentally cutting a wire, power breakdown etc. which can cause major disruptions and some part of system may not be functioning as desired. It is easy to reproduce these types of faults.
Intermittent: These faults occur randomly. Mostly these faults are ignored while testing the system and appear only when the system goes into operation.
Transient: These faults are very common in computer system, caused by the temporary malfunction of the system or due to some external interference. These faults are of short duration and cause an error only in the duration for which they exist. Detection of such fault should be very hard and expensive.

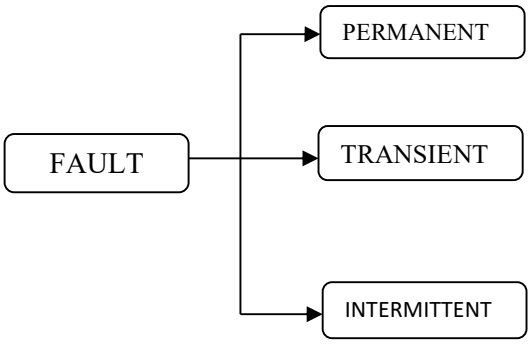

Figure 2: Types of fault in a system.

Due to presence of these faults, the system encounters many problems during execution of a task. This ultimately leads to a **FAULTY ENVIRONMENT.**

ERROR- An undesirable system state that may lead to failure of the system.
FAILURE- Failure occurs when the system can no longer perform its task or execution as required or when a running system deviates from its behavior. There are different types of failures such as: Crash Failure, Omission Failure, Timing Failure, Response Failure, and Arbitrary (byzantine) Failure.

3. FAULT TOLERANCE

Faults leads to the problems in normal functioning of distributed system and may turn system execution in the immoral direction. In air traffic control system, distributed disaster system, railways reservation system, internet banking a single fault may lead to massive loss of money and even human lives. In such a circumstance inclusion of fault tolerance technique is vital. Fault tolerant systems are basically designed to make sure that in the event of an error, crash or failure, data is safe and the machine repeatedly provides its specified services, thereby increasing the reliability of a system. The existing fault tolerance schemes can be generally classified as check-point based, replication based or a hybrid of both. Moreover there are some additional preventive schemes are exception handling, software rejuvenation and third party rendezvous. Each scheme has its own advantages and disadvantages that decide their appropriateness for a given application area. Fault Tolerance is required for better outcome of results, reliable processing and to avoid faults which occur in the system.

4. DESCRIPTION OF DIFFERENT FAULT TOLERANT APPROACHES

Different Fault tolerance techniques are used in Distributed systems as given below:

S.NO.	WORK DONE	COMMENTS
1	Selecting an Optimal Fault Tolerance Strategy for Reliable Service-Oriented Systems with Local and Global Constraints by Zibin Zheng and Michael R. Lyu, Member, IEEE, 2015	In this paper [5] the problem of selecting an optimal fault tolerance technique for reliable service oriented systems is investigated. They make the user requirements as local constraints and form the selection of fault tolerance strategy as an optimization problem. A heuristic algorithm is projected to proficiently solve the optimization problem. For semantically related tasks fault tolerance strategy selection is also investigated in this paper. The simulation results show that the proposed FT-HEU algorithm can provide optimal results with minute computation time.

2	"Using Host Criticalities for Fault Tolerance in Mobile Agent Systems by Rajwinder Singh and Mayank Dave	In this paper [6] they described a mobile agent based fault prevention and detection technique. This technique is determined by calculating weights taking into consideration the criticality of the hosts by their monitoring agents which keep updating the weights of hosts from time to time. These weights are used for decision making of check pointing and the monitoring mobile agents act together to detect undesirable behaviors, also provide support for restoring the system back to normal state. From experimental results it can be safely inferred that the proposed monitoring technique for multi agent distributed application may effectively increase system's fault tolerance along with effectual recognition of damages in system.
3	Antecedence Graph Approach to Check pointing for Fault Tolerance in Mobile Agent Systems by Rajwinder Singh and Mayank Dave, Senior Member, IEEE	In this paper [3] a novel parallel check pointing algorithm antecedence graph approach for achieving fault tolerance in mobile agent systems is described. The proposed scheme utilizes the check pointed information for fault tolerance which is stored in form of antecedence graphs. In case of failures, using check pointed information, the antecedence graphs and message logs are regenerated for recovery and then normal operation continued. In addition compared with the existing schemes, this algorithm involves the least number of mobile agents during the check pointing procedure, which increases the system performance. Moreover the quantitative analysis and numerical results show that the proposed algorithm has better performance than existing ones and the overheads for proposed scheme are significantly low
4	Lightweight Fault-tolerance Mechanism for Distributed Mobile Agent-based Monitoring by Jinho Ahn, IEEE 2014	This paper [10] presents a fault-tolerance method suitable for large scale and dynamic mobile agent-based monitoring organizations. It is capable of detecting quick failure by sending heart-beat messages to its next higher level manager. The various non-faulty managers are minimized which are certainly affected by failure of domain managers. However, in case of agent creation and termination, failure detections are to be initiated continuously. Takeover actions are performed even in simultaneous failures of domain managers.
5	Vinod Kumar Yadav, Mahendra Pratap Yadav and Dharmendra Kumar Yadav , "Reliable Task Allocation in Heterogeneous Distributed System with Random Node Failure",IEEE 2012	In this paper [2] to examine the service reliability of the heterogeneous distributed system they use a two phase hybrid approach. In the first phase candidate nodes for task are determined. In second-phase concept of load sharing is used for the finishing of tasks. If a node failed earlier than executing the tasks assigned on it, it will transfer the remaining tasks to next appropriate candidate node. This phase can repeat until the whole tasks got finished. The simulation result shows that this hybrid solution gives the more cost effective results than by using load sharing approach. For a small test case of eight tasks, it improved the performance up to 20% from load sharing solutions.

6	Adaptive Check pointing for Fault Tolerance in an Autonomous Mobile Computing Grid by Parmeet Kaur Jaggi and Awadhesh Kumar Singh 2014 IEEE	In this paper [7] for failure recovery of mobile nodes in a Mobile computing grid, an adaptive check pointing technique is presented. An adaptive replication based method has been used for saving the checkpoints of application nodes in the grid. The method replicates a node's checkpoint data at multiple nodes, depending on the availability of resources in the mobile computing grid. As survival of any one of the replicas of the checkpoint data will make sure successful recovery. The proposed scheme has been simulated and the results of simulation show a significant improvement in failure recoverability as compared to a check pointing scheme that does not employ replication. .
7	Fault Tolerance in Heterogeneous Distributed Systems by Zhe Wang, Naftaly and H. Minsky,IEEE2014	In this paper [8] a fault tolerance mechanism for handling coordination failures in heterogeneous distributed systems is introduced. At application level the common Fault tolerant techniques require code-injection, which is not feasible for open systems, because the lack of control over the code of the components. To control the flow of messages between system components they propose a coordination mechanism called Law-Governed Interaction (LGI), independent from the code of system components. The coordination failures within an open system can be handled, by providing a range of FT measures that can be established by controlling messaging. In general this method for fault tolerance can be used for Heterogeneous distributed systems. The preliminary testing and experiments of their implementation show that this method is feasible and promising.
8	Fuzzy Rule based Check pointing Arrangement for Fault Tolerance in Mobile Grids by Pritee Parwekar and Parmeet Kaur, IEEE 2014	In this paper [9] to construct a well-organized check pointing arrangement for mobile computing grid, a Fuzzy Rule based system (FRS) is designed. By FRS, those nodes having sufficient resources are selected as Checkpoint Storage Nodes (CSNs) and other mobile hosts transfer their check pointed data to one of the selected neighboring CSN. The aim of FRS is to enhance the probability of recovery of check pointed data subsequent to a failure; in that way allowing a distributed application to complete its execution successfully on the mobile computing grid. Simulation results show a significant improvement in failure recoverability as compared to a random check pointing arrangement.

5. CONCLUSION

From our survey it is clear that, to detect a fault in distributed computing system is a very difficult task. For real time distributed system, there is no chance of a fault, a single fault in these systems leads to failing of the whole system, if not properly detected and recovered at proper time because these systems depends on time. Things which are kept in mind while applying fault tolerance in heterogeneous distributed computing systems are scalability, feasibility and reliability of the task. Reliability can be achieved by updating the task of the faulty node to the active or candidate node. This also increases the system performance. As more and more research is going on in distributed computing , the new fault tolerant techniques such as Fuzzy Rule based Check pointing, FT-HEU algorithm, Adaptive check pointing etc. reduces the execution time of the task and hence increases the performance of system to a great extent.

REFERENCES

[1] "Distributed Computing Principles, Algorithms, and Systems" by Ajay D. Kshemkalyani (University of Illinois at Chicago) and Mukesh Singhal (University of Kentucky, Lexington) © Cambridge University Press 2008, www.cambridge.org.

[2] Vinod Kumar Yadav, Mahendra Pratap Yadav and Dharmendra Kumar Yadav, "Reliable Task Allocation in Heterogeneous Distributed System with Random Node Failure", International Conference of Computing Science, Vol 61, pp: 187-192, 2012.

[3] Rajwinder Singh and Mayank Dave, Senior Member, "Antecedence Graph Approach to Check pointing for Fault Tolerance in Mobile Agent Systems", IEEE transactions on computers, vol. 62, no. 2, February 2013.

[4] Z. Li, X. Liu, W. Ren, and L. Xie, "Distributed tracking control for linear multiagent systems with a leader of bounded unknown input," IEEE Transactions on Automatic Control, vol. 58, no. 2, pp. 518-523,2013.

[5] Zibin Zheng and Michael R. Lyu, "Selecting an Optimal Fault Tolerance Strategy for Reliable Service-Oriented Systems with Local and Global Constraints" IEEE TRANSACTIONS ON COMPUTERS, VOL. 64, NO. 1, JANUARY 2015.

[6] Jinho Ahn, "Lightweight Fault-tolerance Mechanism for Distributed Mobile Agent-based Monitoring" IEEE International Conference on Mobile Agent-based monitoring, 2009.

[7] Parmeet Kaur Jaggi and Awadhesh Kumar Singh, "Adaptive Checkpointing for Fault Tolerance in an Autonomous Mobile Computing Grid, IEEE, 2014.

[8] Zhe Wang, Naftaly and H. Minsky," Fault Tolerance in Heterogeneous Distributed Systems" 10th IEEE International Conference on Collaborative Computing: Networking, Applications and Work sharing (CollaborateCom 2014).

[9] Pritee Parwekar and Parmeet Kaur, "Fuzzy Rule based Checkpointing Arrangement for Fault Tolerance in Mobile Grids", IEEE2014.

[10] Rajwinder Singh, Mayank Dave, "Using Host Criticalities for Fault Tolerance in Mobile Agent Systems, 2nd IEEE International Conference on Parallel Distributed and Grid Computing, Vol 2 pp: 67-72, Jun 2012.

[11] Bahi, Jacques, Couturier, Raphael and Vernier, Flavien. Synchronous distributed load balancing on dynamic networks, Journal of Parallel and Distributed Computing, Elsevier Inc., Vol. 65, Issue 11, 1397 – 1405, 2010.

[12] Pradeep K. Sinha Distributed Operating Systems Concepts and Design, PHI Learning Private Limited, New Delhi- 110001, 2010.

Madhav Gupta is currently pursuing M.Tech from Chandigarh Group of Colleges (CGC) Landran. He received engineering degree in 2013 from Baba Ghulam Shah Badshah University (BGSBU), Rajouri, J&K. His area of interest includes distributed computing, parallel processing and network security.

Dr. Ruchi Singla is working as Professor and Head of Department, Electronics and Communication Engineering, Chandigarh Engineering College, Landran, Mohali. She completed her doctorate from Thapar Univeristy in 2013. Her area of interest includes. Wireless communication and Antenna Designing.