Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 4, Issue. 6, June 2015, pg.601 – 605

REVIEW ARTICLE

Overloading in Distributed Computing System- A Review

Suman Joshi[1], Gagangeet Singh[2]

Student[1], Assistant Professor[2], Department of CSE[1,2]

Chandigarh Group of Colleges, Landran, Punjab

Emails: tanyashahisgood@gmail.com, Cecm.cse.gagangeet@gmail.com

Abstract: Distributed computing utilizes 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. Each system is only aware about the input of the system. So there is a problem of fault tolerance in distributed system when one of the working node become overload. In this paper we have studied about various fault tolerance in distributed system.

Keywords: Computing, Fault Tolerance, openness, reliability, task allocation

1. Introduction

The distributed system is a collection of independent computers that appears to the users as a single coherent system. It can be of two types either homogenous or heterogeneous. A heterogeneous distributed computing system is that where random node can fail permanently. A homogenous distributing computing system is that which shares local memory. Because the DCS is heterogeneous, so its various nodes have different hardware and software characteristics. The different components of the application also have various hardware and software requirements. A distributed system connected by local networks and physically connected with each others. Distributed computing

utilizes 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. A computer program that runs on distributed system is known distributed program. The process of writing such types of languages is called distributed programming [14]. Grid computing and Cluster computing are types of distributed computing systems. A Distributed system consists of a group of independent computers associated through a network and sharing middleware which enables computers to organize their behavior and to share the property of the system so that users identify the system as a single, in corporate computing facility [15]. There are many properties of distributed computing system. First of all each computational entity has local memory. The entities communicate with each others with the help of message passing. Second the system has to tolerate failures in individual computers. The system structure and links may changes during the execution of the distributed programs. Each system is only aware about the input of the system. Resource sharing is the capability to use any hardware, software or data anywhere in the system. Resources in a distributed system, dissimilar the centralized one, are physically encapsulated within one of the computers and can only be accessed from others by communication. Openness is apprehensive with extensions and improvements of distributed systems. New components have to be included with presented components so that the added functionality becomes accessible from the distributed system as a whole [6].

A distributed system as a collection of independent computers that appear to the users of the system as a 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. A distributed system has distinct advantages over a set of non-networked smaller computers [11]. Data can be shared dynamically giving private copies does not work if the data is changing. Peripherals can also be shared. Some peripherals are expensive and/or infrequently used so it is not justifiable to give each PC a peripheral. These peripherals include optical and tape jukeboxes, typesetters, large format color printers and expensive drum scanners. Machines themselves can be shared and workload can be distributed amongst idle machines [3]. Finally, networked machines are useful for supporting person-to-person networking: exchanging email, file transfer, and information access. In distributed system there are some issues. These issues are:

• To Design, implement and using distributed software may be difficult. Issues of creating operating systems and/or languages that support distributed systems arise.

• The network may lose messages and/or become overloaded. Rewiring the network can be costly and difficult.

• Security becomes a far greater concern. Easy and convenient data access from anywhere creates security problems.

2. Review of Literature

In paper [1] tried to solve the problem of maximizing reliability of heterogeneous distributed computing system where random node can fail permanently. They utilize the load sharing policies for handling the nodes failure as well as maximizing the service reliability of DCS. They have undertaken a two-phase hybrid approach to analyze the service reliability of heterogeneous DCS in the presence of communication and node uncertainty. If any processor failed before executing the tasks assigned on it, it will transfer reaming tasks to next more reliable and cost effective processor. Repeat this phase until all the tasks got executed and compared with those derived by load sharing approach. The simulation result shows that, in most of the cases this hybrid solution gives the more cost effective results than load sharing approach. For a small test case of eight tasks, it improved the performance up to 20% from load sharing solutions. Performance graph shows as the number of tasks increases the performance of hybrid approach will also increases. In this paper [2], proposed a Connection Fault- Tolerant Model for mobile environment which considers two communication scenarios first is when MHs can connect to the fixed network through MSS, and the second when MHs cannot connect to the fixed network. They also presented a Decision Algorithm which is responsible for making a decision for a MH when corresponding MH-Ag cannot communicate with its MH for a defined period of time. The CFT model reduces the blocking time of resources at the fixed devices

provides fast recovery from connection failures owing to mobility of mobile devices and increases the number of committed mobile transactions. In this paper [3] presented mobile agent based fault prevention and detection technique where the team of mobile agents monitor each host in mobile agent based system. They proposed an approach to introduce fault tolerance in multi agent system through check pointing based on updating of weights from time to time while calculating the dependence of hosts. 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 beside effective recognition of vulnerabilities in system. In this paper [4] presented 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; therefore we provide an approach which distributes efficiently the information in the network by selecting some objects of the network to be duplicates of information. In this paper [5] they proposed 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 them to stable storage during the normal computation message transmission, the proposed algorithm can reduce the time latency for a global check pointing procedure significantly. The quantitative analysis and numerical results reveal that the proposed algorithm has better performance than existing ones and the overheads for proposed scheme are significantly low. They implementation it by use of other check pointing schemes can also be done to improve the proposed scheme in order to further enhance the execution time and recovery time. In future, work could be done for integrating graph based and non-graph-based schemes to achieve high level of fault tolerance for making real life, mobile agent-based applications more reliable and fault tolerant. In paper [6] Distributed operating systems are still in an early phase of development, with many unanswered questions and relatively little agreement among workers in the field about how things should be done. Many experimental systems use the client-server model with some form of remote procedure call as the communication base, but there are also systems built on the connection model. Relatively little has been done on distributed naming, protection, and resource management, other than building straight- forward name servers and process servers. Fault tolerance is an up-and-coming area, with work progressing in redundancy techniques and atomic actions.

3. Overloading in Distributed System

Overloading means imbalance in the system. Due to overloading fault may occur in the system which is responsible for the degradation of the system.. 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. 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. The real time distributed systems like grid, robotics, nuclear air traffic control systems etc. are highly responsible on deadline. Any mistake in real time distributed system can cause a system into collapse if not properly detected and recovered at time. Fault-tolerance is the important method which is often used to continue reliability in these systems. By applying extra hardware like processors, resource, communication links hardware fault tolerance can be achieved. In software fault tolerance tasks, to deal with faults messages are added into the system.

3.1 Various Techniques for Fault Tolerance: There are various techniques which are used for fault tolerance. These are as follows:

1. **Replicas**

It is well known technique which is used to enhance the availability. It has two strategies either Active or Passive. It helps to remove overhead issues and complexity.

2. Hardware Resilience

It handles network reliability by the hardware unit transparency. It is used for correction also.

3. Check pointing

Fault Tolerance can be achieved through various types of redundancy. Check-point start is the common method. In this method an application starts from the earlier checkpoint after a fault. Application may not be able to meet strict timing targets.

4. Application based Resilience

It deals with faults using information about the application used by it and makes the developer more intelligent.

5. Un co-ordinate check pointing

In this technique, process co-ordinate with their check pointing activities and each process records it local checkpoint independently. It requires cascade rollbacks which lead to the initial state due to domino effect.

6. Task allocation: By using this technique task can be assigned according to weights.

4. Conclusion

Distributed system consists of a group of independent computers associated through a network and sharing middleware which enables computers to organize their behavior. In this paper we focused on various techniques of fault tolerance which can be used to overcome overloading problem.

References

[1] Vinod Kumar Yadav, Mahendra Pratap Yadav and Dharmendra Kumar Yadav , "Reliable Task Allocation in Heterogeneous Distributed System with Random Node Failure: Load Sharing Approach, International Conference of Computing Science, 2012

[2] Tome Dimovski, Pece Mitrevski, "Connection Fault-Tolerant Model for Distributed Transaction Processing in Mobile Computing Environment" *ITI 2011 33rd Int. Conf. on Information Technology Interfaces*, June 27-30, 2011, Cavtat, Croatia

[3] Rajwinder Singh, Mayank Dave, "Using Host Criticalities for Fault Tolerance in Mobile Agent Systems, *2nd IEEE International Conference on Parallel, Distributed and Grid Computing*, 2012

[4] Asma Insaf Djebbar, Ghalem Belalem , "Modeling by groups for faults tolerance based on multi agent systems*", IEEE*,2010

[5] Rajwinder Singh and Mayank Dave, Senior Member, "Antecedence Graph Approach to Checkpointing for Fault Tolerance in Mobile Agent Systems", *IEEE TRANSACTIONS ON COMPUTERS, VOL. 62, NO. 2, FEBRUARY 2013*

[6] W. Qu and H. Shen, "Analysis of mobile agents fault-tolerant behavior", Proceedings of IEEEIWIC/ACM international conference on intelligent agent technology, pp. 377 - 380, 2004.

[7] 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, 2005.

[8] Yang, 1. Cao and W. Wu, "CIC: An integrated approach to checkpointing in mobile agent systems", Proceedings of the Second IEEE International Conference on Semantics, Knowledge and Grid,pp. 1-6, 2006.

[9] J. Philippe, M. Flatin and S. Znaty, Two Taxonomies of Distributed Network and System Management Paradigms, Emerging Trends and Challenges in Network Management, 2000.

[10] A. Liotta , G. Pavlou and G. Knight, Exploiting Agent Mobility for Largescale Network Monitoring, *IEEE Network*, 2002, 7-15.
[11] S. Kwon and J. Choi, An Agent-based Adaptive Monitoring System, *Lecture Notes In Artificial Intelligence, 4088*, 2006, 672-677.
[12] G. Susilo, A. Bieszczad and B. Pagurek, Infrastructure for Advanced Network Management based on Mobile Code, *In Proceedings of the IEEE/IFIP Network Operations and Management Symposium (NOMS'98)*, 1998, 322-333.
[13] Alexandru Costan, Ciprian Dobre, Florin Pop, Catalin Leordeanu, Valentin Cristea, "A Fault Tolerance Approach for Distributed Systems Using Monitoring Based Replication", IEEE, 2010
[14] K. Park, "A fault-tolerant mobile agent model in replicated secure services", *Springer, Proceedings of International Conference Computational Science and Its Applications*, Vol. 3043, pp. 500-509,2004

