International Journal of Computer Applications (0975 – 8887)
Volume 79 – No4, October 2013

Improved Queuing Mechanism for Hybrid Load Balancing Scheme in Interactive Application

Sampada S. Kalmankar
Pune University
Pimpri Chinchwad College Of Engineering, Pune, India

Sudarshan S. Deshmukh
Pune University
Pimpri Chinchwad College Of engineering, Pune, India

ABSTRACT

Distributed interactive applications (DIA) are becoming popular in the recent years. Examples of DIAs include shared workspaces, networked games, distributed whiteboards, distributed architectural design, virtual classrooms, telemedicine and simulation. The essential aspect of DIAs is that sufficient information is communicated between participants so that the state of the application remains consistent for all participants at all times. Consistent refers to the state of all the systems. If nodes have inaccurate information about the state of other nodes, due to communication delays between nodes, this could result in unnecessary periodic exchange of loads among them, due to which, certain nodes may become idle while loads are in transit, this would result in the prolonged total completion time of a load.

Hence load balancing becomes more challenging in interactive applications as load variation is very large and the load on each server may change continuously over time, when the server takes the load migration decision, the load status collected from other servers may not be valid. This will affect the performance, of the load balancing algorithms. All the existing methods neglect the effect of network delay among the servers on the load balancing solutions. In this paper, due to the change in the load of the server, network delay would affect the performance of the load balancing algorithm. A new priority packet scheduling scheme is proposed in which load requesting Interactive application packets are placed in the highest priority queue and the processing of packets at other queue. Simulation results show that the proposed buffered priority packet scheduling scheme outperforms AODV with single queue for the load requesting messages of Interactive application in term of end-to-end data transmission delay.

General Terms

Load balance, efficient wireless protocol AODV.

Keywords

Buffer, Delay, Distributed system, Load balancing, Multilevel Queue, Packet scheduling, Priority Scheduling.

1. INTRODUCTION

A distributed system is a system where the information processing is distributed over several computers rather than confined to a single machine. This allows users to interact among themselves through networks. Applications of DIAs include shared workspaces, Automated Banking Systems, networked games, distributed whiteboards, distributed architectural design, Global Positioning Systems, virtual classrooms, telemedicine and simulation, Tracking Roaming Cellular Telephones, Research and development project.

Distributed Interactive Application or DIA is a group of users connected via a network to interact synchronously with a shared application state. An environment that hosts an application involving cooperation and communication between remote users over a communications network will be called a Distributed Interactive Application or DIA. In these Distributed System Applications load dynamics is much higher than the other applications. These are the class of computer programs that involve multiple simultaneous users located at geographically diverse locations. As a result, performance of Load balancing in delay filled environments depends upon the selection of nodes and the amount of the load that are allowed between nodes to exchange. In the other case where network delays are very large, it would be easier to reduce the amount of load exchange so that the time is not wasted while loads are in transit. As the load dynamics is high for the DIA, the essential aspect of DIAs is that sufficient information is communicated between participants so that the state of the application remains consistent for all participants at all times. Networks have the limitations in terms of bandwidth and latency. Latency is nothing but lag experienced, due to which the real-time interactive experience of users within DIAs is destroyed. One of the experienced limitations is, if nodes have inaccurate information about the state of other nodes, due to communication delays between nodes, this could result in unnecessary periodic exchange of loads among them, due to which, certain nodes may become idle while loads are in transit, this would result in the prolonged total completion time of a load.

Several approaches and techniques have been studied and developed to overcome the effects of latency. Most of the techniques focus on reducing the quantity of data that must be transmitted to maintain consistency between users of the DIA else they focus on the increasing the processing power of the participating nodes and the divulgence of higher network bandwidth.

The motivation the study is to minimize the delay of networks for DIA where there is a high demand for the quicker communication. The attempt to improve the utilization of networks can be hardware and software approaches. In this paper, an approach at the software level is discussed and the delay of the load requesting messages of DIA is minimized. This algorithm is based on the decentralized approach; it can also be applied to the centralized approach. The experimental results of proposed algorithm shows that the proposed schemes can significantly improve the performance by minimizing the delay and also any of the efficient load balancing algorithms implemented on it.

2. RELATED WORK

A number of load-balancing schemes have been developed, considering a variety of resources, including the CPU, memory, disk I/O, or a combination of CPU and memory resources. These approaches have proven effective in increasing the utilization of resources, assuming that network interconnects is not potential bottlenecks [1].There are many researches focusing on the issue of distributed load balancing for CPU and memory resources. Harchol-Balter and Downey [2] developed a CPU-based pre-emptive migration policy that was shown to be more efficient than non pre-emptive migration policies. Zhang et al. [3] studied load-sharing policies that consider both CPU and memory services among the nodes. Even though these schemes can effectively utilize memory and/or CPU resources at each node in the system, nobody have considered the effective usage of I/O or network resources. I/O-aware load balancing scheme is proposed by Xiao Qin to meet the needs of a cluster system with a variety of workload conditions [4]. This approach is able to balance implicit disk I/O load as well as that of explicit disk I/O.

All the above approaches are effective under workloads without high communication intensity and balancing the communication load is not considered. A communication-sensitive load balancer has been proposed by Cruz and Park [5]. In Communication aware load balancing scheme which is different from their work in this scheme attempts to simultaneously balance two different kinds of I/O load, namely, communication and disk I/O [1]. Penmatsa and Chronopoulos [6] took into account communication delay and considered dynamic load balancing with the assumption that all jobs have the same execution time.

Since the repartitioning from scratch using traditional graph partitioning methods is an overhead, methods based on refining existing partitions and migrating extra loads among the processors have been proposed for finite element analysis [7]. These methods minimize the overall overheads. The load migration problem can be represent as an optimization problem with an intention to minimize both the load difference among servers and the amount of load needed to be migrated. The idea that models the incremental graph partitioning problem as a linear programming problem, and optimization-based load balancing method for multi-server DIA is proposed in [8]. In this method the loads of a DVE are represented using a connected graph, where nodes representing a user and edge representing the communication cost between the corresponding two nodes the load migrations are performed among adjacent partitions so as to minimize the imbalance of load among the partitions and the inter-partition communication costs of the reassigned nodes. As it has a very high computational cost.

In decentralized approach is that there are the local servers which are assigned to manage the partitions, these perform the load balancing process. Each server will determine the amount of load to be transferred its neighbour servers. In [9], a method is proposed where when a server is overloaded, it uses the load information of its neighbouring servers to determine the amount of load to transfer to each of the neighbour servers. Hence the load balancing solutions are only be considered as temporary and the overloaded server may quickly become overloaded again. In [10], a method is proposed to address the limitation of [9] by considering more servers for load transfer. If the neighbouring servers are overloaded then cannot transfer its entire extra load hence it will consider the neighbour servers of its available neighbour servers and so on. Although this method may produce better load balancing solutions, it is slower due to the extra communication and computation overheads.

In queuing analysis, there have been some works that consider delays in the design of the load balancing algorithms. In queuing analysis, [11] conduct extensive analysis and reveal that computational delays and load-transfer delays can significantly degrade the performance of load balancing algorithms that do not account for any delays. Further extension of this work is proposed to consider the random arrivals of the external tasks. Both of them try to minimize the task completion time due to network and/or computational delays. On the contrary, [12] they have shown that when the local servers have received the load status from the central server after some network delay, the loads of the local servers may have a different one and the load balancing solutions may no longer be accurate as load dynamics is much higher. The effect of delay in the queue is shown in [13]. A deterministic dynamic nonlinear time-delay system is developed to model load balancing in a cluster of computer nodes used for parallel computations.

Several approaches and techniques are proposed and developed to overcome the effects of latency. Most of these focus on reducing the amount of data that must be transmitted to maintain consistency between users of the DIA. The most widely used algorithm for achieving this is dead reckoning, which predicts the future positions of users over a short term interval [14]. Dead reckoning reduces the number of packets that must be transmitted to maintain state consistency in the following way. Each node uses dead reckoning to determine an entity's position. At the entity's home node, a comparison is performed between the actual entity's position and the position provided by the dead reckoning algorithm. When the difference between these positions exceeds the threshold value an update packet with the correct entity position is generated and transmitted to all other nodes.

In this paper the concept of multi level queuing is used. In [15] it is found that if queue priorities are added in the scheduling intelligently then performance is improved. Data model helps choosing appropriate preferences. In [16], they have proposed a three class priority packet scheduling scheme. Real-time packets are placed into the highest priority queue and can pre-empt the processing of packets at other queues. Other packets placed based on the location into two other queues. Simulation results show that the proposed three-class priority packet scheduling scheme outperforms FCFS and multi-level queue schedulers in terms of end-to-end data transmission delay.

3. PROPOSED WORK

In the First Paper, [1] a behavioral model for parallel applications is introduced with large requirements of network, CPU, and disk I/O resources. Furthermore, they have addressed the issue of improving the effective bandwidth of networks on clusters at the software level without requiring any additional hardware. Specifically, a dynamic communication-aware load balancing scheme, referred to as COM-aware, for non dedicated clusters where the resources of the cluster are shared among multiple parallel applications being executed concurrently. A simple and efficient means to measure communication load imposed by processes has been presented.

From this paper it is concluded that the focus is on the improvement of the network resources as most of the researches focus on the memory, I/O or CPU resources. Most of the researches focus on the usage of bandwidth else the reduction of the data transfer else increased processor speed leaving opportunity to improve the effective bandwidth of networks on clusters running parallel applications.

In the second paper [12], A distributed virtual environment (DVE) allows users to interact with virtual objects or among themselves through networks. As the number of concurrent user's increases, these systems may reach their performance bottleneck and no longer provide the quality of service required, typically in terms of response time. From this paper, the difficulty that due to communication delays among servers is discussed, the load balancing process may be using outdated load information from local servers to compute the load migration data flows, while the local servers may be using outdated balancing flows to conduct load migration, this would significantly affect the performance of the load balancing algorithm. To address this problem, two methods are proposed: uniform adjustment scheme and adaptive adjustment scheme.

In the third paper [13], the main objective of this paper is to analyze the effects of delays in the exchange of information among computational Element's, and the constraints these effects impose on the design of a load balancing strategy. A deterministic dynamic nonlinear time-delay system is developed to model load balancing. The model is shown to be self consistent in that the queue lengths cannot go negative and that the total number of tasks in all the queues and the network is conserved. From this paper it is concluded that there is a need to minimize the queuing delay. Comparison of the above studied papers is as shown below [17].

By studying the above papers the problem statement is that there is a scope to improve the performance of network resource for the Distributed Interactive Applications where the load dynamics is much higher. If the load dynamics on one particular node is more than the load migration time then there is a need to reduce the load dynamics which is due to the property of the interactive application, which may not be possible as this property is inherent property of the interactive application. To know that most of the researchers have done their work on communication load balancing mechanisms but that was implemented for common applications only.

Further for Interactive applications latency should be in lined with the degree of load change, from literature survey study it is found that latency for a network has high impact of software overhead then delay of a network.

Some researches consider the load balancing approaches for interactive applications but none have considered load index as a parameter for load. As per my knowledge, there is no work done on communication aware load balancing for interactive applications. All existing mechanism addresses reduction network delay as a parameter of study but here it is considered that software overhead for reducing it.

3.1 Features

- Load Balancing

The computing power of any distributed system can be realized by allowing its constituent computational elements (CEs), or nodes, to work cooperatively so that large loads are allocated among them in a fair manner. Load distribution among CEs is called load balancing (LB). An effective LB policy ensures optimal use of the distributed resource hereby no CE remains in an idle state while any their CE is being utilized.

- Interactive Application

Distributed Interactive Applications or DIAs are essentially a class of computer programs that involve multiple simultaneous users located at geographically diverse locations who are all connected by a communications network and who cooperate with each other in a shared virtual environment to accomplish a task or set of tasks. The essential aspect of DIAs is that sufficient information is communicated between participants so that the state of the application remains consistent for all participants at all times. The single biggest obstacle to achieving consistency is the very technology that facilitates the DIAs the network itself.

- Latency

Network latency is simply defined as the time delay observed as data transmits from one point to another. Latency is a measure of time delay experienced in a system, and also depends on the system and the time being measured. Latency is measured from the start of exchange of a data unit at the application layer of one participating node to the end of exchange of the same data unit at the application layer of another node.

The total latency can be calculated as below.

$$\tau_{total} = \sum_{i=1}^{n} \tau_{process}^{i} + \sum_{i=1}^{n} \Delta\tau_{propogation}^{i+1} + \sum_{i=1}^{n} \tau_{queue}^{i} + \sum_{i=1}^{n} \tau_{network}^{i} \qquad (1)$$

Where τ_{total} refers to Total latency.

N is the number of nodes, including source and destination nodes;

$\tau_{process}^{i}$ is the processing delay

$\Delta\tau_{propogation}^{i+1}$ is the propagation delay

τ_{queue}^{i} is the queuing delay.

$\tau_{network}^{i}$ is the network delay.

3.2 Constraints & Assumptions

- The load generated is communication specific.
- Network delays for load migration are not considered.
- Latency cannot be reduced beyond certain value.
- Distance between the nodes parameter is not considered.
- Load dynamics cannot be reduced for interactive applications beyond certain level.
- The entire client and the server are considered to be synchronized.

4. PROGRAMMER'S DESIGN

4.1 Key problems in communication

- The communication hardware includes node memory and I/O architecture, network interface and the network.
- The communication software, includes protocol structure and the algorithms

The main focus is on the software overhead. Software overhead dominates communication time. It follows that communication time cannot be significantly reduced, even with very efficient network and NIC, if there is no efficient communication software. The three sources of software overhead are:

- The software needs to traverse several protocol layers. A common technique to reduce this type of overhead is to simplify the protocol structure.

- Message communication may involve a number of memory copying instances, which call for a zero-copy protocol.

- Communication software may cross protection boundaries several times in transmitting a message.

4.2 Latency Reduction Methods

- The packet processing delay can be minimized educed in a number of ways: by reducing the quantity of data on the network, by increasing the processing power at routers and source/destination nodes and by using more efficient processing algorithms.

There is the need to reduce the queuing delay by giving the highest priority to the DI Applications, which will reduce the delay in the queues.

5. PROPOSED ALGORITHM

In the proposed scheme, the idea is that the highest priority IA requests are processed with a minimum possible delay. They are placed in non-pre-emptive priority1 tasks queue and can pre-empt the currently running data. Thus, they are expected not to reserve a queue location for a longer period of time. Moreover, the number of pre-emption's will be low since IA requests are generally small in number. On the other hand, non-real-time packets that arrive from the sensor nodes at lower level are placed in the pre-emptive queue.
IA request Packets are usually processed in FCFS fashion. Each packet has flag as a REQ packet which indicates that it is a request packet and also the node number which indicates the node whose current load is requested. There is a load table maintained at the gateway which consists of the node number and the current load of that particular load. By checking the node number and the current load the gateway will reply with the load on the particular requested node.

5.1 Pseudocode

Algorithm: Two queue priority data scheduling scheme

```
while packetk,i received by gatewayscheduleri at level k i.e., at $l(k)$ do
if packetk, i, type = 0 (i.e lbal type) then
    Buffering i.e two buffers are created one for IA request packet and other for the normal requests.
    if flag=IAR && $node_i$ // $node_i$ indicates the load of i
      is requested
        put packetk, in bp1 queue/buffer // IA request task
        Packets in this buffer1 are considered for
            processing
        Table of loads is accessed $node_i$ load is replied
            back to the requested node.
    else
        put taskk, i, in bp2 queue // non IA request task or
            data packets
    end if
    Assume, the duration of a timeslot at $l(k) \leftarrow t(k)$
     Load requesting time of $node_i$ at $l(k) \leftarrow$
      $loadReqTime_k(t)$
      LoadReplyTime after Requesting,
              $t_1(k) \leftarrow t(k) + Delay_k(t)$
    Let, total load request packets for $node_i$ at time k $l(k) \leftarrow$
              $n_k(bp_1)$
    All these requests are processed at the gateway.
    Simultaneous routing of data packets is done by the
      gateway.
end if
end while
```

5.2 Proposed Architecture

In the proposed architecture will be consisting of the central server and the number of local servers and the number of clients served by the local servers. These clients will be running with the interactive applications whose load will be continuously varying. In between the central server and the local server there will be a gateway in which the proposed scheduling algorithm is implemented. Following are the functions of the scheduler algorithm

- Scheduler needs to decide whether the incoming task is an interactive or a common application.

- Scheduler will maintain two buffers one with the highest priority for the request packets and another for the data packets.

- Scheduler processes the request packet and the data packets are routed further.

Following figure 1 shows the architecture of the proposed system.

6. EXPERIMENTAL RESULTS

AODV uses routing tables, one route per destination, and destination sequence numbers, a method to avoid loops and to determine freshness of routes. By studying the different papers it is seen that AODV is better than the DSR, DSDV, I-DSDV, DYMO, OLSR, ZRP. The average end-to-end delay of a data packet is the time interval when a data packet generated from Constant Bit Rate source completely received to the application layer of the destination.

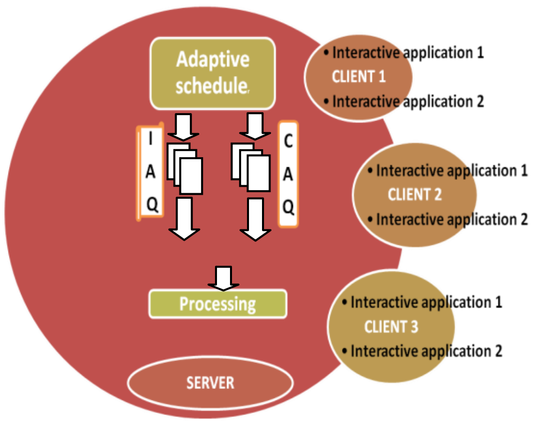
Adaptive schedule
I A Q
C A Q
Processing
SERVER
• Interactive application 1
CLIENT 1
• Interactive application 2
• Interactive application 1
CLIENT 2
• Interactive application 2
• Interactive application 1
CLIENT 3
• Interactive application 2

Fig 1: Architecture of the Proposed System

AODV outperforms when compared with the DSDV and I-DSDV as shown in the above graph Fig 7. When the AODV is compared with the other routing protocols i.e ZRP,OLSR, DSR and DYMO even then the AODV outperforms in terms of End-to-End Delay. This means that there is a minimum delay when AODV protocol is used. This is shown in the below fig 2.

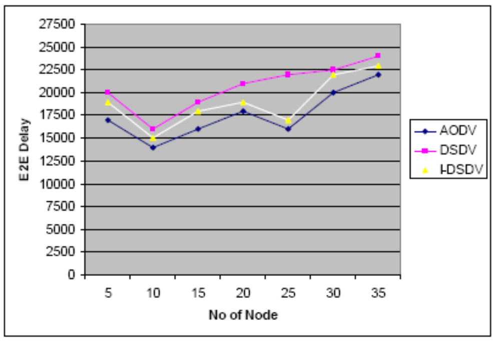

Fig 2: End to End Delay

In [18] it is shown that AODV didn't produce so much delay even the number of nodes increased and also if there is varying speed, AODV produces less End to End Delay. As seen in the above results the delay is end to end minimum for AODV, the performance of AODV protocol is improved by inserting a new queue. In this work It is considered that initially 20 clients and the gateway of the server and one server.

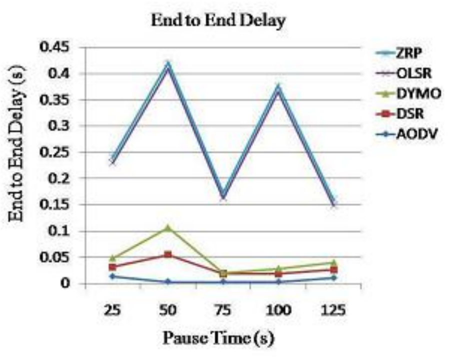

Fig 3: End to End Delay

The gateway have got 2 buffers one will be dedicated to the interactive application load requests and the other one will be for the rest of the requests coming in which are to be routed to the server. The IA requests coming in are processed by the gateway itself by accessing a table stored on it which consists of the load status.

Simulations on Network simulator is carried out and have defined the parameters for the performance evaluation of AODV and LBAL routing protocols under different pause time while the number of nodes is fixed. The simulation parameters are summarized in table I. The pause time are set to 0, 30, 60, 90, 120, 150, 180,210,240 and 270 second.

6.1 Performance Matrix:

In actual real time there are only 20% of the tasks are of load requesting. Hence only 20% of the load requesting tasks are generated. These requests consists of the node number of which the load is requested, the type of the message whether it is a requesting or the data transfer.

Here a new protocol named as "LBAL" is implemented, which actually is a improved AODV for DIA. Here the performance is measured by executing the simulation for 25 times. Whenever a node requests for the load of the other node at t then the time spent for the reaching the requested load may be t+d where d is the delay. During this time if the load is changed then the information got by the node will be incorrect so the DIA system may be inconsistent.
Hence there is a need to reduce the delay such that the load status received by the requester and actual load is same.

Firstly AODV is considered as a routing protocol and then the LBAL as a routing protocol. The results which are got showed 83% improvement then AODV with the consideration of the load requested and the actual load. Following Table III shows the results. Here when request1 is arrived for requesting the load of some other node the load that particular instance was 1, using LBAL the load responded was also 1, whereas using AODV the load responded was 2. Hence there might be a inconsistent state as the load migrations takes place using these load status. This goes on till number of requests here only few requests are tabulated.
This improvement is because there is no queuing delay as it is minimized almost to 0.

	LBAL		**AODV**	
Requests	**Load requested**	**Actual load**	**Load requested**	**Actual load**
1	1	1	1	2
2	46	46	48	63
3	47	47	1	64
4	47	47	64	127
5	47	47	47	64
6	14	14	128	127
7	128	128	63	158
8	146	146	379	422
9	144	144	231	294
10	245	245	400	463

Following Fig 4 and 5 show the graphs of the load status for AODV and LBAL.

The average end-to-end delay of a data packet is the time interval when a data packet generated and the request is received. It is seen that the end to end delay is zero then delay increases as the traffic increases. A table is maintained at the gateway which is updated at regular intervals. Here updation is done at every 0.6sec for AODV. For LBAL It is set to 0.6 then it was too slow hence it is reduced to 0.3 by which results are better.

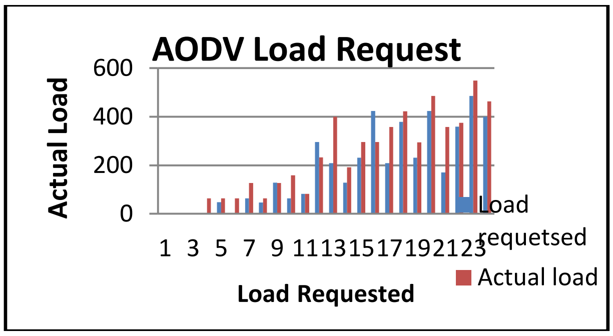

Fig 4: AODV Load Status

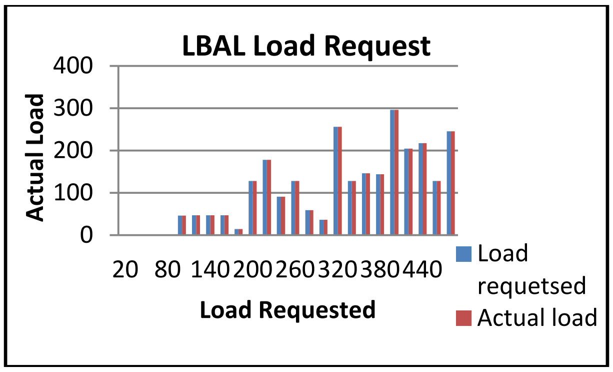

Fig 5: LBAL Load Status

Since the delay is minimized, It is seen that throughput is also increased by 83%. This can be seen in below fig 6.

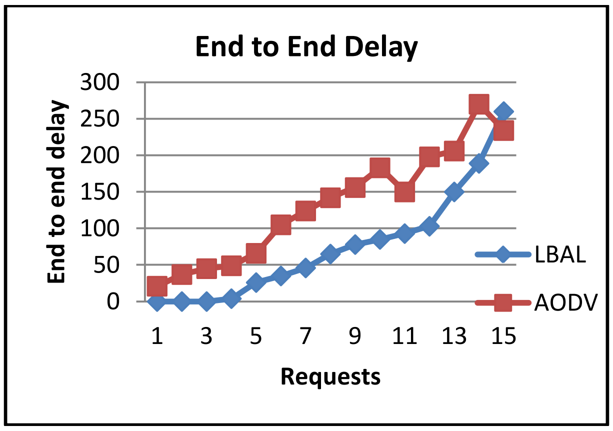

Fig 6: End to End Delay

As the end to end delay is minimized the turnaround time is also minimized. It is seen that when compared to AODV turnaround is minimized and then there is more than 100% improvement. By this It is concluded that whenever the load request is sent and response received is same using LBAL protocol. Due to which the load balancing algorithm ensured that the load is migrated to the under loaded node only, hence it consistency is maintained.

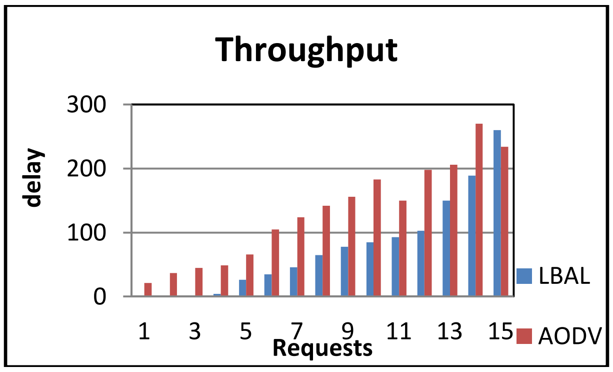

Fig 7: Throughput

7. CONCLUSION

Due to communication delays among servers, the load balancing process may be using outdated load information to conduct load migration. This would significantly affect the performance of the load balancing algorithm and also create inconsistency in the system as the load balancing decisions are taken based on these updates.

For the interactive applications latency should be in lined with the degree of load change, from literature survey study it is seen that latency for a network has high impact of software overhead then delay of a network. Hence in proposed algorithm the queuing delay is minimized by giving the highest priority to the load requesting Interactive applications. The proposed algorithm improves the performance by 83% that is delay is minimized.

8. REFERENCES

[1] Xiao Qin, Hong Jiang, Adam Manzanares, Xiaojun Ruan and Shu Yin, IEEE "*Communication-Aware Load Balancing for Parallel Applications on Clusters*" IEEE TRANSACTIONS ON COMPUTERS, VOL. 59, NO. 1, JANUARY 2010.

[2] M. Harchol-Balter and A.B. Downey, "*Exploiting Process Lifetime Distributions for Dynamic Load Balancing*," ACM Trans. Computer Systems, vol. 15, no. 3, pp. 253-285, 1997.

[3] I. S. X.-D. Zhang, L. Xiao and Y.-X. Qu, "*Improving Distributed Workload Performance by Sharing Both CPU and Memory -Resources*," Proc. 20th Int'l Conf. Distributed Computing Systems (ICDCS '00), pp. 233-241, 2000.

[4] Xiao Qin, Hong Jiang *Improving Effective Bandwidth of Networks on Clusters using Load Balancing for Communication-Intensive Applications*, Proceedings of the 24th IEEE International Performance, Computing, and Communications Conference (IPCCC 2005).

[5] J. Cruz and K. Park, "*Towards Communication-Sensitive Load Balancing*," Proc. 21st Int'l Conf. Distributed Computing Systems, pp. 731-734, Apr. 2001.

[6] Satish Penmatsa and Anthony T. Chronopoulos,"*Dynamic Multi-User Load Balancing in Distributed Systems*", 1-4244-0910-1/07/$20.00 c 2007 IEEE.

[7] Y. Hu, R. Blake, and D. Emerson. *An optimal migration algorithm for dynamic load balancing*. Concurrency: Practice and Experience, 10(6):467–483, 1998.

[8] J. Lui and M. Chan. *An efficient partitioning algorithm for distributed virtual environment systems*. IEEE Trans. on Parallel and Distributed Systems, 13(3):193–211, 2002.

[9] B. Ng, A. Si, R. Lau, and F. Li. *A multi-server architecture for distributed virtual walkthrough*. In Proc. ACM VRST, pages 163–170, 2002

[10] K. Lee and D. Lee. *A scalable dynamic load distribution scheme for multi-server distributed virtual environment systems with highly-skewed user distribution*. In *Proc. ACM VRST*, pages 160–168, 2003.

[11] J. Douglas Birdwell, J. Chiasson, Z. Tang, C. Abdallah, M. Hayat, and T. Wang. *Dynamic time delay models for load balancing. Part I: Deterministic models*. In Proc. CNRS-NSF Workshop: Advances in Control of Time-Delay System, 2003.

[12] Yunhua Deng and Rynson W.H. Lau. *On Delay Adjustment for Dynamic Load Balancing in Distributed Virtual Environments*, IEEE TRANSACTIONS ON

VISUALIZATION AND COMPUTER GRAPHICS, VOL. 18, NO. 4, APRIL 2012.

[13] John Chiasson, Zhong Tang, Jean Ghanem, Chaouki T. Abdallah, J. Douglas Birdwell, Majeed M. Hayat, and Henry Jérez, "*The Effect of Time Delays on the Stability of Load Balancing Algorithms for Parallel Computations*", IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY, VOL. 13, NO. 6, NOVEMBER 2005

[14] Aaron McCoy, Tomás Ward, Seámus McLoone and Declan Delaney, "*Formalizing a Framework for Dynamic Hybrid Strategy Models in Distributed Interactive Applications*" IEE Irish Signals and Systems Conference, Dublin, June 28-30, 2006.

[15] Diwakar SHUKLA, Shweta OJHA, Saurabh JAIN "*Data Model Approach And Markov Chain Based Analysis Of Multi-Level Queue Scheduling*", Journal of Applied Computer Science & Mathematics, no. 8 (4) /2010, Suceava

[16] Lutful Karim, Nidal Nasser, Tarik Taleb, and Abdullah Alqallaf, "*An Efficient Priority Packet Scheduling Algorithm for Wireless Sensor Network*"

[17] Sudarshan Deshmukh and Sampada S Kalmankar "*Comparative Study of Effects of Delay in Load Balancing Scheme for Highly Load Variant Interactive Applications*", Proc. of Int. Conf. on Advances in Communication, Network, and Computing 2013

[18] Abdul Hadi Abd Rahman, Zuriati Ahmad Zukarnain "*Performance Comparison of AODV, DSDV and I-DSDV Routing Protocols in Mobile Ad Hoc Networks*", European Journal of Scientific Research ISSN 1450-216X Vol.31 No.4 (2009), pp.566-576