Hybrid Approach for Effective Load Balancing of Distributed File Systems in Cloud Computing

A. Lalitha, P. Mahendrababu, Prof. T. Rajendran

Abstract- Load balancing plays an effective role on performance in cloud computing environment. Efficient load balancing makes cloud computing more efficient and improves performance of the entire system. In our existing system achieves load balancing with the help of load rebalancing algorithm using DHT protocol.But the existing solutions are not efficient for load balancing so that the various methods have been developed to resolve new problems. In our proposed system introduces a partitioning method for storing static and dynamic files using partitioning method in various situations for efficient load balancing.AVL tree algorithmused along with a maze technique to improve the system performance and reduces the time complexity.

Index Terms- A Maze Technique, Cloud Computing, Load Balancing, Partitioning.

1 INTRODUCTION

Cloud computing is an effective and well-chosen technology in the field of computer science. Distributed technologies in cloud computing will satisfied the user needs and they provide various applications. The cloud is changing our life by providing users with different types of services. NIST gave a definition of cloud computing as a model for enabling convenient, ubiquitous, on-demand network access to a shared pool of configurable computing resources (e.g., applications, networks, storage, servers, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. More and more people give attention to cloud computing.

Cloud computing is more efficient and scalable. But, maintaining the several jobs in this environment is a complex problem with load balancing. The job arrival pattern is not predictable and capacities of each node in the cloud differ, for load balancing problem. When the environment is larger in size and also more complex, these partitions simplifies the load balancing process.

In this proposed system have the algorithm that chooses the correct partitions for arriving files while the node balances for each partition chooses the effective load balancing strategy. In this system we are interested to studying the load balancing problem in cloud environment specialized for large-scale and dynamic system. Such a system has more number of nodes with different behavior. All the nodes and files are more dynamic in nature. Our objective is to find out the path where the static and dynamic files are taken place and balance the loads of node. Additionally, we aim to reduce the time delay and then the performance will be increased.

- *A.Lalitha is currently pursuing master degree program in computer science and engineering in Chettinad College of Engineering and Technology, Karur, Tamil Nadu, India. E-mail: lalitaarjun91@gmail.com*
- *P.Mahendrababu is working as Assistant professor in department of computer science and engineering in Chettinad College of Engineering and Technology, Karur, Tamil Nadu, India. E-mail: mahenbabu@gmail.com*
- *Prof T. Rajendran is working as the Head of Department of Computer Science and Engineering and Information Technology in Chettinad College of Engineering and Technology, Karur, Tamil Nadu, India. E-mail: rajendran_tm@yahoo.com*

2 RELATED WORK

Earlier studies have many of the load balancing algorithms in cloud computing like, Round Robin Algorithm, Throttled Load Balancing Algorithm, Load Rebalancing Algorithm, Equally Spread Current Execution Algorithm, etc. These all the algorithms are try to achieve higher throughput, smaller the response time and also avoid overload. Here, we can discuss all the algorithms that are used in existing system.

Round Robin Algorithm is based on the random sampling method. This algorithm randomly selects the load when the system is unbalanced. Unbalanced system can involve both overload and under load stage. This algorithm is not suitable for all kind of system.

Throttled Load Balancing Algorithm is fully based on virtual machine. Initially, client will make a request to the load balancer to find a Virtual Machine which is suitable to perform the required operation.

Load Rebalancing Algorithm is the one which performing the prior load balancing. This improves the performance of the system, reduces the traffic and movement cost. But this is not applicable for all the situations.

Equally Spread Current Execution Algorithm is using the load balancer to spreading the loads into different virtual machines. Based on the priority the loads are distributed randomly to the virtual machines. Before the distribution process it will check the size of the file and load capacity of virtual machine. This will increases the throughput and also take less time.

DHT Protocol is the one which makes every node to be decentralized and rebalance the load by itself. But it requires more computing power and movement cost. So we can move on to the proposed system.

3 PROPOSED SYSTEM

Load balancing schemes depending on whether the system dynamics are important can be either static or dynamic. Static schemes do not use the system are less complex while dynamic schemes will bring the additional costs for the system but can change as the system status changes.

The load balancing is aimed at the cloud which has both static and dynamic nodes. The cloud will chooses the suitable partitions for arriving jobs while the partitioning technique for each node partition chooses the best load balancing strategy. In our proposed system can uses the some of the techniques like a maze technique and also partitioning. These are all the techniques will be discussed furthermore.

3.1 A Maze Technique

A maze is a technique which is mainly used by the files to find out their exact suitable node. The existing system using the migration scheme instead of that we can proposed the maze techniques that will reduces the time complexity.

The maze technique can have two values (0 and 1).Where value 0 represents the node is static and a value of 1 represents the node is dynamic. If the files can be separated then our system complexity will be reduced. Then the maintenance of the system is to be simple.

We may have the chance to go in different directions. Not knowing which one to select but save our current position. With each new location we will examine the new possibilities.

Sometimes we can search the path that is already visited. These will takes the additional time to find out the exact position. To avoid the visiting same path again, we can mark the node as already visited. Then the node will not searching the same node again. And then it reduces the time complexity.

3.2 Partitioning Technique

The load balancing in cloud is still an existing problem that needs new architecture to overcome the complexities. And the load balancing plays an effective role on improving the overall performance and reducing time delay. This partitioning technique will discuss below.

```
//finding the overloaded node
procedurenode (i, V)
    j← V[i]
    if j<0 then return(i)
    elsereturn(node(j))
endif
endprocedure

// finding overloaded node and compress V
procedurenode(i,V)
    j←v[i]
    if j <0 then return(i)
    elseV[i]←node(j);
    return(V[i])
endif
endprocedure

//joining two file partitioning together
procedurejoin-filepartition(i,j,V)
    ri ←node(i,V); rj ←node(j,V);
    Ifri!=rjthen
        si ← - V[ri];
    sj ←-V[rj]
        ifsi<sjthen
            V[i] ←rj;
            V[j] ←-(si+sj)
        else
            V[j] ←ri;
            V[i] ←–(si+sj)
        endif
    endif
    return (V)
endprocedure
```

3.3 System Design

Our system design is discussed given below.

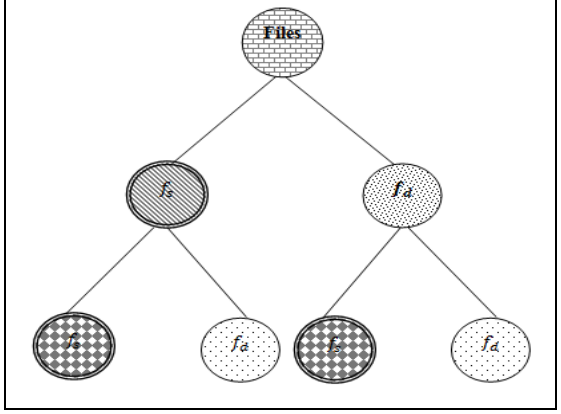

Fig. 1.System Design

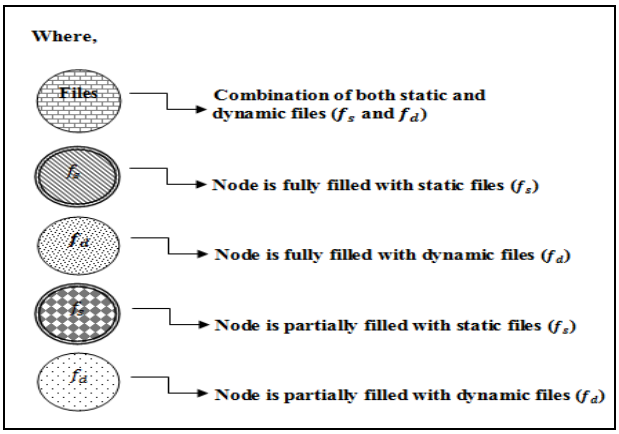

Fig. 2.Description of Nodes

3.4 Procedure

//Initialize the list to the maze entrance coordinates and direction{left or right}

While(list is not null) {
(i,j,dir)= coordinates and direction from end of list;
While(there are more moves from (i,j){
(g,h)= coordinates the next move;
AvlNode<K,E>*y=new AvlNode<T>(k,e);
if(k<pp→key)pp→leftChild=y;
//insert left
else pp→rightChild=y;
// insert right
//Adjust balance factors of all nodes.
//all nodes on path presently have a balance factor of 0.Balance factor d = +1 then k is inserted in left subtree of node a. Or d = -1 then k is inserted in the right subtree of node a.
int d; AvlNode<K,E>*b,*c;
if(k>a→key) {
b =p=a→rightChild; d=-1; }
while(p!=y)
if(k>p→key)
p→bf=-1;p=p→rightChild;
}else
{p→bf=1; p=p→leftChild;
}// *is tree unbalanced?*
if(!(a→bf)||!(a→bf+d))
{*//tree still balanced*
a→bf+=d; return;}
//tree unbalanced, determine the rotationtype
if(d= =1)
{*//left imbalance*
if(b→bf==1)
{*//rotation type LL*
a→leftChild=b=b→right Child;
b→rightChild=a; a→bf=0; b→bf; }
else{
//rotation type LR
c =b→rightChild;
b→rightChlid=c→leftChild;
a→leftChild=c→rightChild;
c→leftChild=b; c→rightChild=a;
switch (c→bf) {
Case1: a→bf= -1; *//dynamic*
b→bf=0; break;
Case -1: b→bf = 1; *//static*
a→bf = 0; break;
Case 0: b→bf= 0; a→bf=0;
break;}
c→bf=0; b=c;
}}
else{
//right imbalance}
//subtree with node b has been rebalanced
if(!pa)root=b;
elseif(a==pa→leftChild)
Pa→leftChild=b;
else pa→rightChild=b;
return;}
if((fs==f))&&((fd==f))success;
}}

By using the above algorithm in our system we can balance the loads effectively these will surely improving the performance of our system and reducing the time complexity.

4 EXPECTED OUTCOME

Comparing with the Fig.3 and 4, the overall system performance is improved.

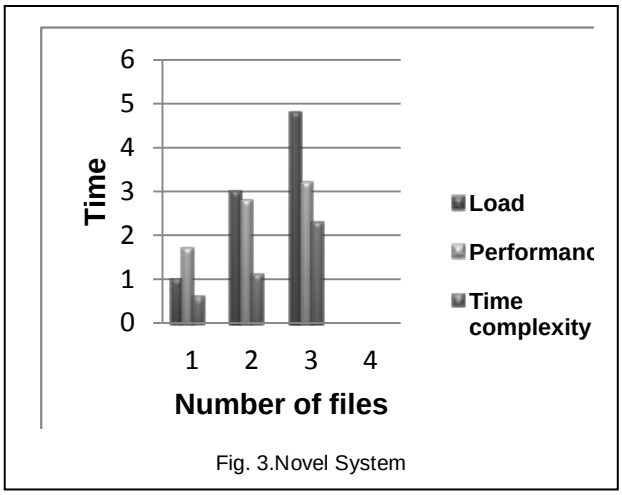

Fig. 3.Novel System

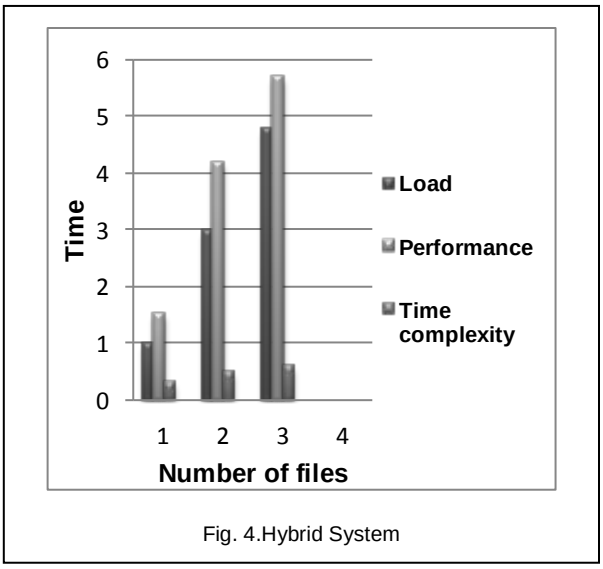

Fig. 4.Hybrid System

It shows that, the static and dynamic files will be stored in balanced state. The graph strongly shows the hybrid model will provide higher performance than the novel system and it also reduces the time complexity.

5 CONCLUSON

The novel load balancing system to deal with the problem of load balancing in dynamic and distributed file system. In that system, load balancing is not more effective in-terms of movement cost, network traffic, time delay and load imbalance. So in our proposed system introduced the partitioning method for storing static and dynamic files in various situations for efficient load balancing. The effective load balancing will be achieved using a maze and partitioning technique. Our system aims to develop the enhanced strategies through hybrid model for load balancing. The proposed load balancing system outperforms in-terms of load imbalance, time complexity and system performance.

ACKNOWLEDGMENT

The authors are grateful to the anonymous reviewers who have provided us with valuable comments to improve their study.

REFERENCES

[1] Hung-Chang Hsiao, Member, IEEE Computer Society, Hsueh Yi Chung, HaiyingShen, Member, IEEE, and Yu-Chang Chao, "*Load Rebalancing for Distributed File Systems in Clouds*" IEEE Trans. Parallel and Distributed Systems, vol. 24, no.5, May. 2013.

[2] Hadoop Distributed File System Rebalancing Blocks, http:// developer.yahoo.com/hadoop/tutorial/module2.html# rebalancing, 2012

[3] I. Stoica, R. Morris, D. Liben-Nowell, D.R. Karger, M.F. Kaashoek, F. Dabek, and H. Balakrishnan, *Chord: A Scalable Peer-to-Peer Lookup Protocol for Internet Applications*, IEEE/ACM Trans. Networking, vol. 11, no. 1,pp. 17-21, Feb. 2003.

[4] M. A. Weiss, *Data Structures and Algorithm Analysis*, Benjamin Cummings, RedwoodCity, California, 1994.

[5] G. DeCandia, D. Hastorun, M. Jampani, G. Kakulapati, A.Lakshman, A. Pilchin, S.Sivasubramanian, P. Vosshall, and W.Vogels, "*Dynamo: Amazon's Highly Available Key-Value Store*,"Proc. 21st ACM Symp. Operating Systems Principles (SOSP '07),pp. 205-220, Osct. 2007.

[6] J.W. Byers, J. Considine, and M. Mitzenmacher, "*Simple Load Balancing for Distributed Hash Tables*," Proc. First Int'l Workshop Peer-to-Peer Systems (IPTPS '03), pp. 80-87, Feb. 2003.

[7] Y. Zhu and Y. Hu, "*Efficient, Proximity-Aware Load Balancing forDHT-Based P2P Systems*," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 4, pp. 349-361, Apr. 2005.

[8] H.C. Hsiao, H. Liao, S.S. Chen, and K.C. Huang, "*Load Balance with Imperfect Information in Structured Peer-to-Peer Systems*," IEEE Trans. Parallel Distributed Systems, vol. 22, no. 4, pp. 634-649,Apr. 2011.

[9] L.M. Ni, C.W. Xu, and T.B. Gendreau, "*A Distributed DraftingAlgorithm for Load Balancing*," IEEE Trans. Software Eng., vol. 11,no. 10, pp. 1153-1161, Oct. 1985.

[10] M. Parashar and J.C. Browne, "*On PartitioningDynamic Adaptive Grid Hierarchies*", Proceedings of the 29th Annual Hawaii International Conference on System Sciences, Jan. 1996.

[11] Tushar Desai, JigneshPrajapati, "*A Survey Of Various Load Balancing Techniques And Challenges In Cloud Computing*", International Journal Of Scientific & Technology Research Volume 2, Issue 11, November 2013.

[12] Y. Zhu and Y. Hu, "*Efficient, Proximity Aware Load Balancing for DHT-Based P2P Systems*," IEEE Trans. Parallel and Distributed Systems, vol. 16, no. 4, pp. 349-361, Apr. 2005.