Load Balancing Model for Distributed File System in Cloud Computing using Distributed Hash Table Approach

Ms. Arpita M. Hirudkar Dr.V.M.Thakare Dr. R.V Dharaskar

Abstract **:- Load balancing in the cloud computing environment has an important impact on the performance. Good load balancing makes cloud computing more efficient and improves user satisfaction. The Cloud has become a popular data storage provider and users are relying heavily on it to keep their data. Furthermore, most cloud data servers replicate their data storage infrastructures and servers at various sites to meet the overall high demands of their clients and increase availability. The technique, Dual-Direction FTP utilizes the availability of replicated files on distributed servers to enhance file download times through concurrent downloads of file blocks from opposite directions in the files. Distributed file systems are key building blocks for cloud computing applications. In such file systems, nodes simultaneously serve computing and storage functions; a file is partitioned into a number of chunks allocated in distinct nodes. The load rebalancing issue in distributed file systems specialized for large-scale, dynamic and data-intensive clouds. Files can also be dynamically created, deleted, and appended. This results in load imbalance in a distributed file system; that is, the file chunks are not distributed as uniformly as possible among the nodes. A fully distributed load rebalancing algorithm is used to solve the load imbalance problem. The main aim of proposed dynamic load balancing technique for the distributed file system is to reallocate file blocks such that the blocks can be distributed to the system as uniformly as possible while reducing the movement cost as much as possible and to enhance download speed and reduce the concurrent transfer overhead.**

Keywords **– cloud computing, load balancing, DHT.**

I. INTRODUCTION

The Cloud environment has characteristics distinct from most other distributed environments: First, the resources available are geographically distributed over huge areas. Second, the resources are heterogeneous, thus having various platform architectures, operating environments and devices; also, the resources are connected through shared dynamic and heterogeneous communication infrastructures; and the different components in the environments in some situations. If several Cloud servers have replicas of the files, only one of them will be serving a client. Therefore, load balancing techniques rely on distributing the multiple clients' requests among the servers to reduce waiting time for each client; however, this does not improve the total download time per client since it has to be done through a single connection on a single server. The cloud is changing our life by providing users with new types of services. Users get service from a cloud without paying attention to the details. cloud computing is a technology for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources like networks, servers, storage, applications, and services that can be rapidly provisioned and released with minimal management effort or service provider interaction. Key enabling technologies for clouds include the MapReduce programming paradigm, distributed file systems, virtualization, and so forth. These techniques emphasize scalability, so clouds can be large in scale, and comprising entities can arbitrarily fail and join while maintaining system reliability. Distributed file systems are key building blocks for cloud computing applications based on the MapReduce programming paradigm. Cloud computing is an attracting technology in the field of computer science.

II. BACKGROUND

The original FTP was designed to support file transfer in a distributed environment using the client/server model and serving a single connection at a time. Most cloud data servers replicate their data storage infrastructures and servers at various sites to meet the overall high demands of their clients and increase availability. However, most of them do not use that replication to enhance the download performance per client. To make use of this redundancy and to enhance the download speed, here introduce a fast and efficient concurrent technique for downloading large files from replicated Cloud data servers and traditional FTP servers as well. The Dual-Direction FTP (DDFTP) technique is use to enhance download speed and reduce the concurrent transfer overhead. The idea is based on a basic concept of no synchronization parallelization. Instead of alternating block transfer from the beginning of the file to be downloaded, which will require continuous monitoring and control and may not achieve good load balancing, the transfer starts from either end of the file and continue until the servers meet somewhere in the middle.

The load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data-intensive clouds. Load rebalancing algorithm aim to reduce network traffic (or movement cost) caused by rebalancing the loads of nodes as much as possible to maximize the network bandwidth available to normal applications. It also suggest offloading the load rebalancing task to storage nodes by having the storage nodes balance their loads spontaneously. This eliminates the dependence on central nodes. The objective is to design a load rebalancing algorithm to reallocate file chunks such that the chunks can be distributed to the system as uniformly as possible while reducing the movement cost as much as possible. Here, the movement cost is defined as the number of chunks migrated to balance the loads of the chunkservers. The chunkservers are organized as a distributed hash tables (DHT) network; that is, each chunkserver implements a DHT protocol.

There are several cloud computing categories with work focused on a public cloud. A public cloud is based on the standard cloud computing model, with service provided by a

service provider. A large public cloud will include many nodes and the nodes in different geographical locations. Cloud partitioning is used to manage this large cloud. A cloud partition is a subarea of the public cloud with divisions based on the geographic locations. Thus, load balancing model divides the public cloud into several cloud partitions. When the environment is very large and complex, these divisions simplify the load balancing. The cloud has a main controller that chooses the suitable partitions for arriving jobs while the balancer for each cloud partition chooses the best load balancing strategy.

III. PREVIOUS WORK DONE

Earlier the dual direction technique is used to enhance large message transfers using multiple network interface cards (NICs) [4]. Later it tried to offer a way to enhance overall data download times from FTP servers through parallelization across available mirrored sites, while maintaining efficient load balancing with minimum overhead [5]. The original FTP as described in RFC 959 [6] was designed to support file transfer in a distributed environment using the client/server model and serving a single connection at a time. Most cloud data servers replicate their data storage infrastructures and servers at various sites to meet the overall high demands of their clients and increase availability. However, most of them do not use that replication to enhance the download performance per client. To make use of this redundancy and to enhance the download speed, here introduce a fast and efficient concurrent technique for downloading large files from replicated Cloud data servers and traditional FTP servers as well. The Dual-Direction FTP (DDFTP) technique is use to enhance download speed and reduce the concurrent transfer overhead. The idea is based on a basic concept of no synchronization parallelization [1].

Earlier HDFS federation [7] suggests architecture with multiple namenodes. Its file system namespace is statically and manually partitioned to a number of namenodes. However, as the workload experienced by the namenodes may change over time and no adaptive workload consolidation and/or migration scheme is offered to balance the loads among the namenodes, any of the namenodes may become the performance bottleneck. The load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data-intensive clouds. Load-balancing algorithms based on DHTs have been extensively studied [8]. However, most existing solutions are designed without considering both movement cost and node heterogeneity and may introduce significant maintenance network traffic to the DHTs. In contrast, the proposal not only takes advantage of physical network locality in the reallocation of file chunks to reduce the movement cost but also exploits capable nodes to improve the overall system performance [2].

Previous studies have shown that the load balancing strategy for a cloud partition in the normal load status can be viewed as a non- cooperative game. The job arrival pattern is not predictable and the capacities of each node in the cloud differ, for load balancing problem, workload control is crucial to improve system performance and maintain stability. [9] Introduced the tools and techniques commonly used for load balancing in the cloud. The load balancing model based on cloud partition is aimed at the public cloud which has numerous nodes with distributed computing resources in many different geographic locations [3]. Thus, given model divides the public cloud into several cloud partitions. When the environment is very large and complex, these divisions simplify the load balancing [10].

IV. ANALYSIS AND DISCUSSIONS

Dual-Direction FTP (DDFTP) technique, load-balancing algorithms based on DHT (Distributed Hash Table) and the load balancing model based on cloud partition, are used to achieve the load balancing in cloud computing.

1. Dual-Direction FTP (DDFTP) technique.

The Dual-Direction FTP (DDFTP) technique use to enhance download speed and reduce the concurrent transfer overhead. The idea is based on a basic concept of no synchronization parallelization. Instead of alternating block transfer from the beginning of the file to be downloaded, which will require continuous monitoring and control and may not achieve good load balancing, it starts the transfer from either end of the file and continue until the servers meet somewhere in the middle. Thus there will be no need for constant monitoring or reallocation of the load during the download. In addition, the DDFTP servers will not need to be aware of any synchronization or coordination efforts. Those will be present and executed locally on the DDFTP client. DDFTP allows to distribute the download efforts thus reducing the restrictions imposed by the TCP flow and error controls and allowing the client to fully utilize whatever bandwidth available to it by accepting multiple flows simultaneously. At the same time DDFTP makes use of the reliable in-order delivery features of TCP to ensure ordered delivery of the blocks; thus, eliminating the need to use block numbers to order the blocks thus further reducing the overhead.

2. Load-balancing algorithms based on DHT (Distributed Hash Table).

The main objective is to design a load rebalancing algorithm to reallocate file chunks such that the chunks can be distributed to the system as uniformly as possible while reducing the movement cost as much as possible. Here, the movement cost is defined as the number of chunks migrated to balance the loads of the chunkservers. The chunkservers in proposal are organized as a DHT network; that is, each chunkserver implements a DHT protocol. A file in the system is partitioned into a number of fixed-size chunks, and "each" chunk has a unique chunk handle (or chunk identifier) named with a globally known hash function such as SHA1. The hash function returns a unique identifier for a given file's pathname string and a chunk index. Each chunkserver also has a unique ID. Load rebalancing algorithm introduced when each node has global knowledge of the loads of all nodes in the system. Based on the global knowledge, if node i finds it is the least-loaded node in the system, i leaves the system by migrating its locally hosted chunks to its successor i + 1 and then rejoins instantly as the successor of the heaviest node (say, node j). To immediately relieve node j's load, node i requests min{ Lj – , A } chunks from j. That is, node i requests A chunks from the

heaviest node j if j's load exceeds 2A; otherwise, i requests a load of Lj - A from j to relieve j's load. Node j may still remain as the heaviest node in the system after it has migrated its load to node i. In this case, the current least-loaded node, say node i', departs and then rejoins the system as j's successor. That is, i' becomes node j + 1, and j's original successor i thus becomes node j + 2. Such a process repeats iteratively until j is no longer the heaviest. Then, the same process is executed to release the extra load on the next heaviest node in the system. This process repeats until all the heavy nodes in the system become light nodes.

3. The load balancing model based on cloud partition.

The load balancing model based on cloud partition model integrates several methods and switches between the load balance methods based on the system status. A relatively simple method can be used for the partition idle state with a more complex method for the normal state. The load balancers then switch methods as the status changes. Here, the idle status uses an improved Round Robin algorithm while the normal status uses a game theory based load balancing strategy.

a. Main controller and balancers

The load balance solution is done by the main controller and the balancers. The main controller first assigns jobs to the suitable cloud partition and then communicates with the balancers in each partition to refresh this status information. The balancers in each partition gather the status information from every node and then choose the right strategy to distribute the jobs.

b. Assigning jobs to the nodes in the cloud partition

When the load status of a cloud partition is normal, this partitioning can be accomplished locally. If the cloud partition load status is not normal, this job should be transferred to another partition. When a job arrives at the public cloud, the first step is to choose the right partition. The cloud partition status can be divided into three types:

(1) Idle: When the percentage of idle nodes exceeds α, change to idle status.

(2) Normal: When the percentage of the normal nodes exceeds β, change to normal load status.

(3) Overload: When the percentage of the overloaded nodes exceeds γ , change to overloaded status.

When job i arrives at the system, the main controller queries the cloud partition where job is located. If this location's status is idle or normal, the job is handled locally. If not, another cloud partition is found that is not overloaded. The algorithm is shown in Algorithm 1

```
begin
   while job do
          searchBestPartition(job);
          if partitionState = = idle || partitionState = = normal
then
               Send Job to Partition;
          else
               search for another Partition;
          end if
    end while
end
```

Algoorithm 1. Best Partition Searching

Comparison and Effect of outcome of various Attributes and parameters

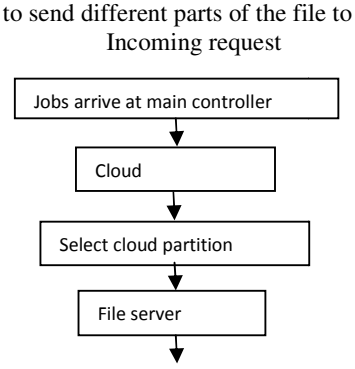
Table 1 shows the effect and comparison of various parameters for Dual-Direction FTP (DDFTP) technique, Load-balancing algorithms based on DHT (Distributed Hash Table) and the load balancing model based on cloud partition. The CPU utilization is maximum using DDFTP technique.

Techniques	Parameters				
	Partition	Bandwidth	Download Time	Movement Cost	CPU Utilisation
DDFTP	File partition-blocks	High 95-97 mbps	66.5s	____	maximum
LB algorithm based on DHT	File partition-chunks	100 mbps	____	Low	maximum
LB model based on cloud partition.	Cloud partition	____	Minimum	Low	Maximum

Table 1

V. PROPOSED METHODOLOGY

The main aim of proposed cloud based load balancing technique using distributed hash table approach for the distributed file system is to reallocate file blocks such that the blocks can be distributed to the system as uniformly as possible while reducing the movement cost as much as possible and to enhance download speed and reduce the concurrent transfer overhead. In distributed file systems, a constant number of replicas for each file blocks are maintained in distinct nodes (file servers) to improve file availability with respect to node failures and departures. These distinct nodes should have the equal or least load to achieve the good load balancing .At the initial stage task request arrive at the main controller then controller selects the cloud partition to assign the job to the file server nodes using best partition searching algorithm. The file servers are organized as a distributed hash tables which implements the DHT protocol. To download the file block from the distinct nodes, the proposed technique gives the most obvious approach is to divide the file and allow multiple servers to send different parts of the file to the client.



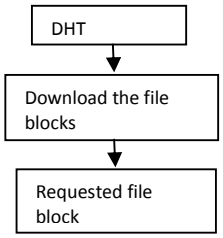


Fig. 1 Flow diagram of cloud based load balancing technique using distributed hash table approach for the distributed file system.

VI. POSSIBLE OUTCOME AND RESULTS

The proposed dynamic load balancing method balance the loads of nodes and reduced the demanded movement cost as much as possible. Also offers an effective way to parallelize file downloads from replicated servers which reduce the download time and provides an efficient load balancing that allows minimizing overhead and server idle time during the file transfer.

VII. CONCLUSION

A novel load-balancing algorithm to deal with the load rebalancing problem in large-scale, dynamic, and distributed file systems in clouds. DDFTP uses the concept of processing the files two different directions. Thus if there are two replicas of a file on two servers one server will send blocks starting at the beginning of the file, while the second will start from the end. They will continue until they meet and the client will then ask them both to stop. A Load Balancing Model Based on Cloud Partitioning introduces a better load balance model for the public cloud based on the cloud partitioning concept with a switch mechanism to choose different strategies for different situations. The proposed dynamic load balancing method balance the loads of nodes, reduced the demanded movement cost as much as possible and the download time.

VIII. FUTURE SCOPE

The future work on improving the technique and incorporating more advanced features to automate some of the steps and increase the usability and transparency of the operations.

REFERENCES

[1]. Gaochao Xu, Junjie Pang, and Xiaodong Fu, "A Load Balancing Model Based on Cloud Partitioning for the Public Cloud", *IEEE Tsinghua Science and Technology*, VOL 18, NO. 1, PP 34-39, February 2013.

[2]. Hung-Chang Hsiao, Hsueh-Yi Chung,Haiying Shen and Yu-Chang Chao, "Load Rebalancing for Distributed File Systems in Clouds," *IEEE Transactions On Parallel And Distributed Systems*, VOL 24, NO. 5, PP 951-962, May 2013.

[3]. Nader Mohamed, JameelaAl-Jaroodi and AbdullaEid, "A dual-direction technique for fast file downloads with dynamic load balancing in the Cloud", *Elsevier ScienceDirect Journal of Network and Computer Applications*, VOL 36, NO. 4, PP 1116-1130, July 2013.

[4]. Mohamed N, Al-Jaroodi J, Jiang H, Swanson D. "High-performance message striping over reliable transport protocols". *Journal of Supercomputing* 2006;38(3): 261–78 Springer.

[5]. Mohamed N, Al-Jaroodi J. "Self-configured multiple-network-interface socket". *Journal of Network and Computer Applications* 2010;33(1):35–42.

[6]. Wwwlink.Filetransferprotocol,RFC959,/http://www.faqs.org/rfcs/rfc959. htmlS; viewedNovember2012b.

[7]. Hadoop Distributed File System, http://hadoop.apache.org/hdfs/, 2012.

[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]. B. Adler, Load balancing in the cloud: Tools, tips and techniques, http://www.rightscale. com/info center/whitepapers/Load-Balancing-in-the-Cloud.pdf, 2012.