IJETST- Vol.||02||Issue||06||Pages 2686-2688||June||ISSN 2348-9480 2015

Open access Journal International Journal of Emerging Trends in Science and Technology

A Simple Load Rebalancing algorithm to rebalance the loads in clouds

Authors
Lakshmi urs S M[1], Dr. C.D Guruprakash[2]
[1]Dept. of Computer Science and Engineering, Shridevi Institute of Engineering and Technology, Tumkur, India
Email: *lakshmi.0290@gmail.com*
[2]Professor and Head, Dept of Computer Science and Engineering, Shridevi Institute of Engineering and Technology, Tumkur, India
Email: *cdguruprakash@gmail.com*

Abstract
Distributed file systems are key building blocks for cloud computing applications based on the MapReduce programming paradigm. In such file systems, the nodes are simultaneously serve this computing and storage functions; a file is partitioned into a number of chunks allocated in distinct nodes so that this MapReduce tasks can be performed in parallel over the nodes. However, in a cloud computing environment, failure is the norm, and nodes may be upgraded, replaced, and added in the system. 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. Emerging distributed file systems in production systems strongly depend on a central node for chunk reallocation. In this paper, a fully distributed load rebalancing algorithm is presented to solve with the load imbalance problem. This algorithm is compared against a centralized approach in a production system and a competing distributed solution presented in the literature. The simulation results indicate that our proposal is comparable with the existing centralized approach and considerably outperforms the prior distributed algorithm in terms of load imbalance factor, movement cost, and algorithmic overhead.

Keywords: *Load balance, distributed file systems, clouds,filechunks,load*

INTRODUCTION

Cloud computing is a compelling technology. In these technology, the clients will distribute their assets in demand without fulfilling employment and manages the assets. There are some of the main technologies that is used for clouds that consists the Map Reduce programming, distributed file system, virtualizati0n technique and so on. key enabling technologies includes scalability, so that this can be used in huge scale, and entities that can be used randomly that fails and join while maintaining the system that is free from errors or mistakes.

The MapReduce programming is an method that is used for generating an large sets of data that is based on parallel distributed algorithm on clusters. In this systems, a file is divided into a many of chunks files that is distributed in the nodes that are distinct so that this tasks that is map reduce can be achieved simultaneously over all the nodes.

In this technology, a cloud divides the file interested in a huge amount of incoherent and constant pieces or the chunk files and assign these files into varieties of storage nodes in clouds that is chunk servers. Each of the storage node in clouds the it will finds the frequency of each word that is unique by searching and parsing its chunks files. In these distributed file system, the load of the nodes is relative to the amount of files chunks that the node influences. In this system the files in a cloud can be vigorously created, removed, and attached, and these nodes can be upgraded, replaced and allocated in the file system, the file chunks are not distributed uniformly among all the nodes.

LITERATURE SURVEY

1. **IEEE published in 2004: "Simple Efficient Load Balancing Algorithms for Peer-to-Peer Systems.**

Abstract:

Load balancing is a vital issue of resourceful function of peer-to-peer system. In this topic an two new load-balancing protocols that is used to improve the performance that will proves the performance that are assurance are inside a invariable factor .this new load balancing protocols uses the constant hashing the data structure that will uses the Chord peer to peer network. This two load balancing protocols maintain nearest-optimal data migration cost and Chord's logarithmic query time. the consistent hashing is an hashing method that uses the DHT pattern for giving the nodes of the items in a peer-to-peer network, this objects and nodes are achieved to a familiar space adress, and this nodes will accumulate all the items that resided close by in the address space.

The first protocol steadiness the distribution of the address space of the key to the nodes, which uses a balanced system load if the distributed hash table achieving these items "arbitrarily" into the space address. so from this knowledge, this will proceeds to the first peer to peer system concurrently achieve O(log n) look-up cost, O(log n) degree,and load balance constant-factor load.

The second protocol that proceeds to balance the items that is distributed among all the nodes. This will uses when the allocation of items in the space address are not randomized.

2. **IEEE published in 2011: Load Balance with Imperfect Information in Structured Peer-to-Peer Systems**

Abstract:

The conception of an essential servers,the peers will be execute in an assortment of,structured peer-to-peer network host that has unusual records of an essential servers, and by migrating this servers, so the loads are balanced by peers that are relative to their capacities.

The presented and decentralized the load balance algorithms that is considered the networks that are varied and the organized peer to peer networks that either by implicitly construct the assisting system to influence overall information or absolutely insist the Peer to peer substrates ordered in a organized manner. In this topic, a load balancing algorithm that is single in that each participate peer that is influences on the incomplete facts of the network to approximate the possibility of the circulation of the capacities of peers and the virtual servers load, that will results an inadequate information of the status of the system network.

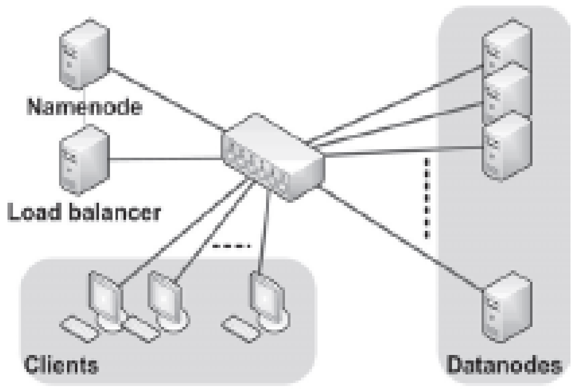
LOAD REBALANCING ALGORITHM

The load rebalancing problem in distributed file systems specialized for large-scale, dynamic and data- intensive clouds to allocate the chunks of files as uniformly as possible among tchunks and aims to reduce network traffic (or movement cost).

We consider a large-scale distributed file system consisting of a set of chunkservers V in a cloud, where the cardinality of V is | V |=n. Typically, n can be 1,000, 10,000, or more. In the system, a number of files are stored in the n chunkservers. First, let us denote the set of files as F. Each file f €F is partitioned into a number of disjointed, fixed- size chunks denoted by C f . For example, each chunk has the same size, 64 bytes, in Hadoop HDFS [4]. Second, the load of a chunkserver is proportional to the number of chunks hosted by the server [3]. Third, node failure is the norm in such a distributed system, and the chunkservers may be upgraded, replaced and added in the system. Finally, the files in F may be arbitrarily created, deleted, and appended. The net effect results in file chunks not being uniformly distributed to the chunkservers.

Our objective in the current study is to design a load rebalancing algorithm to reallocate file chunks such that the chunks can be distributed to the system as uniform 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 chunk servers. Let A be the ideal number of chunks that any chunkserver i € V is required to manage in a system-wide load-balanced state, A node is light if the number of modules it hosts is smaller than the threshold as well as, a heavy node manages the number of modules greater than threshold. A large-scale distributed file system is in a load-balanced state if each module server hosts no more than A modules. In our proposed algorithm, each server node in module I first estimate whether it is under loaded (light) or overloaded (heavy) without global knowledge. This process repeats until all the heavy nodes in the system become light nodes.

SYSTEM ARCHITECTURE



CONCLUSION

A novel load-rebalancing algorithm to deal with the load rebalancing problem in large-scale, dynamic, and distributed file systems. Proposed System will balance the loads of nodes and reduce the demanded movement cost as much as possible, while taking advantage of physical network locality and node heterogeneity.

Proposal is comparable to the centralized algorithm in the Hadoop HDFS production system. Proposed Load-balancing algorithm exhibits a fast convergence rate. Our proposal is comparable to the centralized algorithm in the Hadoop HDFS production system and dramatically out- performs the competing distributed algorithm in terms of load imbalance factor, movement cost, and algorithmic overhead. Particularly, our load-balancing algorithm exhibits a fast convergence rate.

REFERENCES

1. J. Dean and S. Ghemawat, “MapReduce: Simplified Data Processing on Large Clusters,” Proc. Sixth Symp. Operating System Design and Implementation (OSDI ’04), pp. 137-150, Dec. 2004.
2. S. Ghemawat, H. Gobioff, and S.-T. Leung, “The Google File System,” Proc. 19th ACM Symp. Operating Systems Principles (SOSP’03), pp. 29-43, Oct. 2003.
3. Hadoop Distributed File System, http://hadoop.apache.org/hdfs/, 2012.
4. VMware, http://www.vmware.com/, 2012.
5. Xen, http://www.xen.org/, 2012.
6. Apache Hadoop, http://hadoop.apache.org/, 2012.
7. Hadoop Distributed File System “Rebalancing Blocks,” http://developer.yahoo.com/hadoop/tutorial/module2.html#rebalancing,2012.
8. K. McKusick and S. Quinlan, “GFS: Evolution on Fast-Forward,”Comm. ACM, vol. 53, no.3, pp. 42-49, Jan. 2010.
9. HDFS Federation, http://hadoop.apache.org/common/docs/r0.23.0/hadoop-yarn/hadoop-yarn-site/Federation.html, 2012.