Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 85 (2016) 598 – 605

Procedia Computer Science

International Conference on Computational Modeling and Security (CMS 2016)

Data-local Reduce Task Scheduling

Geetha J[a], ***N UdayBhaskar***[a,]*, ***P ChennaReddy***[a,]*

[a]*Department of Computer Science and Engineering, M S Ramaiah Institute of Technology, Bangalore, 560054, India*

Abstract

Inspired by the victory of Apache's Hadoop this paper suggests a new reduce task scheduler. Hadoop is an open source implementation of Google's MapReduce framework. Programs which are written in this functional style are automatically executed and parallelized on a large cluster of commodity machines. The details how to partition the input data, setting up the program's for execution across a set of machines, handling failures of machine and managing the necessary inter-device communication is taken care by runtime system. In the current versions of Hadoop, the map tasks are scheduled with respect to the locality of their inputs in order to shrink network traffic and improve performance. On the other hand, the reduce tasks are scheduled without taking into consideration data locality leading to ruin the performance at requesting nodes. In this paper, we use data locality that is natural with reduce tasks. To accomplish the same, we schedule them on nodes that will result in least amount data- local traffic. Experimental results signify an 11-80 percent decrease in the number of bytes shuffled in a Hadoop cluster.

Keywords: MapReduce, Hadoop , Reduce Task Scheduling , Data-Locality , Rack-Locality.

1. Introduction

Every day, the Stock Exchange generates every day about one terabyte of new trade data [3]. Facebook generates 5 Billion terabytes of data every day. Any such data that cannot be placed into a database falls under the category of unstructured data. As an entity's data footprint grows, the amount of unstructured data to be handled becomes enormous. This is a major cause for several problems .First, where will all this Big Data, as it is being calledtoday, be stored. Second, how do we access all this data, process and analyze the contents. Third, how do we ensure that this data is safe from disk failures, computer failures and so on?

These problems are well-dealt by Hadoop; a reliable, scalable, distributed computing platform developed by

*Corresponding author. Tel.:+919885238374
E-mail address: udaynagella@gmail.com, pcreddy1@rediffmail.com

doi:10.1016/j.procs.2016.05.226

Apache [5].It is an open source implementation of Google's MapReduce framework that allows for the distributed Processing of huge data sets transversely clusters of computers using simple programming models.It is designed to level up from single datacenter to thousands of machines, each offering local computation and storage. At the application layer the library is designed to detect and handle failures, Instead of relying on hardware to deliver high-availability, so a highly-available service on top of a cluster of computers can be delivered each of which may be prone to failures Page Layout.

Hadoop has an underlying storage system called HDFS-Hadoop Distributed file system. To process the data in HDFS, Ha doop provides a MapReduce engine that runs on top of HDFS. This engine has master-slave architecture. The master node is called the JobTracker and the slaves are called TaskTrackers. MapReduce jobs are automatically parallelized across a large set of TaskTrackers. The JobTracker splits the job into several maps and reduce tasks. Hadoop divides the input to the job into fixed size pieces called input splits. The outputs of the map tasks are stored in the local disk. This intermediate output serves as input to the reduce tasks. The consolidated output of the reduce task is the output of the MapReduce job and is stored in HDFS.

Recently, the non-profit organization Spamhaus, which publishes spam blacklists, faced one of the most powerful cyber-attacks seen. This led to cyberspace congestion which affected the Internet overall. The providers of Spamhaus as well as certain Tier 1 Internet Exchanges were the victims of the attack. Cloudflare dubbed it as the attack "that almost broke the Internet." These incidents do highlight the need to develop more efficient security measure to combat such attacks but it also indicates how uncontrolled network congestion can handicap the Internet as a whole. Measures must be taken to tackle any source of congestion within a network. With companies like Facebook, Amazon, AOL and The NY Times using the Hadoop framework which in turn is a cluster of computers connected via a LAN or MAN connection, there is a need to handle any areas that could result in network traffic and hence result in significant delay in providing services to the end user.

A typical Hadoop cluster is a set of computers connected via LAN connection. In case of companies like Facebook, Yahoo, Amazon these clusters consist of thousands of nodes. These Hadoop workers are located in different data centres that are present in geographically dispersed locations thereby avoiding the domino effect that is prevalent when all nodes are connected by a single LAN. A typical MapReduce job may use nodes across data centres .Each node has a local disk, it is efficient to move data processing operations to nodes where application data are located. If data is not available locally in a processing node, data has to be migrate using network interconnects to the node that performs the operation of data processing s. Migrating huge amount of data across data centers and in some cases within data centers may lead to excessive network congestion especially when petabytes of data have to be processed.

In Hadoop, scheduling reduce tasks results in severe congestion problems. The reason being Hadoop task scheduling is based on a 'pull strategy'. The JobTracker does not push map and reduce tasks to TaskTrackers but rather TaskTrackers pull them by making requests. Every TaskTrackers sends a periodic heart beat message requesting a map or reduce tasks. When the JobTracker schedules map tasks, it takes care to ensure that the task runs on a TaskTracker which contains the needed input split. As a result Hadoop MapReduce is said to be data local when scheduling map tasks. On the other hand Hadoop simply schedules any yet-to-run reduce task on any requesting TaskTracker regardless of the TaskTracker's network location [3].This has an adverse effect on network traffic .

This paper proposes scheduling data local reduce tasks. The method ensures that reduce tasks run on a suitable TaskTracker such that there is minimum bytes travelling across the network. Thus, there is a controlled use of the clusters network bandwidth. We have implemented our method on Hadoop 1.0.3.

The rest of this paper is organized as follows. A background on Hadoop MapReduce is given in Section 2. Our proposal and implementation is presented in Section 3, and a performance evaluation in Section 4.Lastly we indicate future works and conclude the paper in Section 5.

.