International Journal of Current Engineering and Technology
E-ISSN 2277 – 4106, P-ISSN 2347 - 5161
©2014 INPRESSCO®, All Rights Reserved
Available at http://inpressco.com/category/ijcet

Research Article

Improving Efficiency of GEO-Distributed Data Sets using Pact

Kirtimalini N. Kakade[Ȧ*] and T.A.Chavan[Ȧ]

[Ȧ]Department of Information Technology,Smt.KashibaiNawale College of Engineering , University Of Pune,Pune ,Maharashtra,India

Accepted 01 May 2014, Available online 01 June, **Vol.4, No.3 (June 2014)**

Abstract

In an Internet era, a report says every day 2.5 quintillion bytes of data is created. This data is obtained from many sources such as sensors to gather climate information, trajectory information, transaction records, web site usage data etc. This data is known as Big data. Hadoop is only scalable that is it can reliably store and process petabytes. Hadoop plays an important role in processing and handling big data It includes MapReduce – offline computing engine, HDFS – Hadoop Distributed file system, HBase – online data access.Map Reduce functions as dividing input files into chunks and processing these in a series of parallelizable steps., mapping and reducing constitute the essential phases for a Map Reduce job. As this freamework provides solution for large data nodes by providing distributed environment. Moving all input data to a single datacenter before processing the data is expensive. Hence we concentrate on geographical distribution of geo-distributed data for sequential execution of map reduce jobs to optimize the execution time. But it is observed from various results that mapping and reducing function is not sufficient for all type of data processing. The fixed execution strategy of map reduce program is not optimal for many task as it does not know about the behavior of the functions. Thus, to overcome these issues, we are enhancing our proposed work with parallelization contracts. These contracts help to capture a reasonable amount of semantics for executing any type of task with reduced time consumption. The parallelization contracts include input and output contract which includes the constraints and functions of data execution The main aim of this paper is to discuss various known Map reduce technology techniques available for geodistributed data sets by using different techniques. Further, the paper also discloses the implementation of these techniques, their advantages, disadvantages, and the results measured. Future trends including use of query optimizing techniques to improve the results of the query as well as reduce the cost for the computation. To achieve this we use the indexing mechanism to the cache system to preserve the query search results.

Keywords: *Geodistributed , MaReduce, PACT, big data*

1. Introduction

Million of users today uses various computer applications Internet services. The sheer volume of data that these services work with has led to interest in parallel processing on commodity clusters. For this the best example is Google, which uses its MapReduce framework to process 20 petabytes of data per day. These services generate clickstream data from millions of users every day, which is a potential gold mine for understanding access patterns and increasing ad revenue Other Internet services, such as e-commerce websites and social networks,also cope with enormous volumes of data.. Furthermore,for each user action, a web application generates one or two orders of magnitude more data in system logs, which are the main resource that developers and operators have for diagnosing problems in production.

In ad hoc parallel processing of arbitrary data Map reduce model is very attractive Map Reduce breaks a computation into small tasks that run in parallel on multiple machines, and scales easily to very large clusters of inexpensive commodity computers. Mapping and reducing include is the main task of Map reduce function.In the initial stage Mappers processes read respective input file chunks and produce <key,val> pairs known to be as “intermediate data”.Each key assigned afterwords to reducer for processing.Hadoop is a free, Java-based programming framework that supports the processing of large data sets in a distributed computing environment. It is part of the Apache project sponsored by the Apache Software Foundation. Hadoop makes it possible to run applications on systems with thousands of nodes involving thousands of terabytes. Its distributed file system facilitates rapid data transfer rates among nodes and allows the system to continue operating uninterrupted in case of a node failure. This approach lowers the risk of catastrophic system failure, even if a significant number of nodes become inoperative. Hadoop was inspired by Google's MapReduce, a software framework in which an application is broken down into numerous small parts. Any of these parts (also called fragments or blocks) can be run on any node in the cluster. Doug Cutting, Hadoop's creator, named the framework after his child's stuffed toy elephant. The current Apache Hadoop ecosystem consists

*Corresponding author: **Kirtimalini N. Kakade** of the Hadoop kernel, MapReduce, the Hadoop distributed file system (HDFS). The Hadoop framework is used by major players including Google, Yahoo and IBM, largely for applications involving search engines and advertising. The preferred operating systems are Windows and Linux but Hadoop can also work with BSD and OS X.

2. HadoopMapreduce

To process huge data sets within an acceptable amount of time, parallel-executable programs are demanded. However, developing such distributed programs are nontrivial and time consuming. For example, it is challenging to organize data and code, and handle failures for such programs. To simplify these tasks, an abstraction to organize parallelizable tasks, MapReduce, is developed. With MapReduce, programmers are only required to develop two functions: 1) Map() for data processing, and 2) Reduce() for collecting and digesting data. MapReduce takes care of nodes coordination, data transport, and failures. Map/Reduce is a "programming model and an associated implementation for pro- cessing and generating large data sets". The Hadoop framework takes the output from the Mapper and does the following –

1. Partitions the output
2. Sorts the individual partitions
3. Sends relevant partitions to Reducers
4. Merges the partitions received from different Mappers
5. Groups the tuples in the partition based on key and calls the reduce function.

3. Related Work And Comparative Analysis

Improving MapReduce Performance in Heterogeneous Environments MapReduce is emerging as an important programming model for large-scale data-parallel applications such as web indexing, data mining, and scientific simulation. A new scheduling algorithm, Longest Approximate Time to End (LATE) is designed, that is highly robust to heterogeneity. we address the problem of how to robustly perform speculative execution to maximize performance. Hadoop's scheduler starts speculative tasks based on a simple heuristic comparing each task's progress to the average progress. Although this heuristic works well in homogeneous environments where stragglers are obvious, we show that it can lead to severe performance degradation when its underlying assumptions are broken. We design an improved scheduling algorithm that reduces Hadoop's response time by a factor of 2. The goal of speculative execution is to minimize a job's response time. Response time is most important for short jobs where a user wants an answer quickly, such as queries on log data for debugging, monitoring and business intelligence. [Chamikara Jayalath et al, 2013]
Volley An Automated Data Placement for Geo-Distributed [S. Agarwal et al, 2010] Cloud Services As cloud services grow to span more and more globally distributed datacenters, there is an increasingly urgent need for automated mechanisms to place application data across these datacenters. We present Volley, a system that addresses these challenges. Cloud services make use of Volley by submitting logs of datacenter requests. Volley analyzes the logs using an iterative optimization algorithm based on data access patterns and client locations, and outputs migration recommendations back to the cloud service. To utilize Volley, applications have to log information on the requests they process. These logs must enable correlating requests into "call trees" or "runtime paths" that capture the logical flow of control across components, as in Pinpoint or X-Trace . Volley takes approximately 14 hours to run through one month's worth of log files. We first map each client to a set of geographic coordinates using the commercial geo-location database mentioned earlier. Iteratively Move Data to Reduce Latency. Volley iteratively moves data items closer to both clients and to the other data items that they communicate with. This iterative update step incorporates two earlier ideas: a weighted spring model as in Vivaldi and spherical coordinates as in Htrae . After computing a nearly ideal placement of the data items on the surface of the earth, we have to modify this placement so that the data items are located in datacenters, and the set of items in each datacenter satisfies its capacity constraints. Like Phase 2, this is done iteratively: initially, every data item is mapped to its closest datacenter. For datacenters that are over their capacity, Volley identifies the items that experience the fewest accesses, and moves all of them to the next closest datacenter. Because this may still exceed the total capacity of some datacenter due to new additions, Volley repeats the process until no datacenter is over capacity.

HOG: Distributed HadoopMapReduce on the GridMapReduce is a framework pioneered by Google for processing large amounts of data in a distributed environment. Hadoop is the open source implementation of the MapReduce framework. Due to the simplicity of its programming model and the run-time tolerance for node failures. The architecture of HOG is comprised of three components. The first is the grid submission and execution component. In this part, the Hadoop worker nodes requests are sent out to the grid and their execution is managed. The second major component is the Hadoop distributed file system (HDFS) that runs across the grid. And the third component is the MapReduce framework that executes the MapReduce applications across the grid.

Existing Mapreduce System for Geodistributed System

There have been many efforts to improve the efficiency of the MapReduce job. Other storage systems like RAMCloud can efficiently handle large amounts of data but have to be deployed within a datacenter. Other System includes a programming model and a framework for developing massively scalable distributed applications. Some system introduce approximation algorithms that order MapReduce jobs to minimize overall job completion time. we describe the model of geo-distributed systems, data, and operations as considered in this paper. First define a partition size to keep our solution bounded. To move data from stage *s to next stage s + 1 a* MapReduce phase is applied to data partitions and the same number of (output) data partitions are created. introduce our solution for optimizing the execution of a MapReduce job

Table I. Comparison of various techniques

Sr.No.	Paper Name	Advantages	Disadvantages
1	Volly is system for automatically geodistributeing data based on the needs of an application	Data migration between data centers will maximaize the efficiency	Cannot control massive data
2	Walter is key value storage system which hold geodistributed data.	Support geodistributed data in cloud environment.	Cannot be deployable in data centers
3	COPS is storage system which supports geodistributed data.	Efficiently handle data which is geographically distributed	Do not support actual computations over stored data.
4	Drayd which is programming model and framework for deploying massive scalable distributed applications.	Provides more fine grained control over execution of geodistributed data	It does not support process mechanism to process geo distributed data.
5	HOG modifies the Hadoop for deployment in the Open Science grid	HOG contribute to use hadoop's map reduce task sheduler similar to hog can process large amount of cross data center intermediate data copy operation	It does not support multiple cloud collaborating with data centers.
6	G-MR mechanism and DTG algorithm in the base paper to maintain and process geodistributed data in cloud environment	G-MR is a mechanism works on geographically distributed data and efficiently provides the solution for optimizing execution of Map reduce job sequence on a given data set .G-MR significantly improves processing time and cost for deosistributed data	More efficient way to process query in efficient way by using PACT Algorithm for more complex jobs.

sequence on a given dataset. Executing individual MapReduce jobs in each datacenter on corresponding inputs and then aggregating results is defined as MULTI execution path.

Drawbacks

• The functions map and reduce alone are not sufficient to express many data processing tasks both naturally and efficiently.
• Map/reduce ties a program to a single fixed execution strategy, which is robust but highly suboptimal for many tasks.
• Map/reduce makes no assumptions about the behavior of the functions. Hence, it offers only very limited optimization opportunities.

4. Need for Solution

The PACT programming model is a generalization of MapReduce, providing additional second-order functions, and introducing output contracts.It provide the efficiency for the more in geodistributedenvironment.In recent years a variety of approaches for web-scale data analysis have been proposed. All of those efforts base on large sets of shared-nothing servers and a massively-parallel jobexecution. However, their programming abstractions and interfaces differ significantly.TheMapReduce programming model and execution framework are among the first approaches for data processing on the scale of several thousand machines. The idea of separating concerns about parallelization and fault tolerance from the sequential user code made the programming model popular. As a result, MapReduce and its open source implementation Hadoop [Had] have evolved as a popular parallelization platform for both industrial and academic research.

Phase - I introduces G-MR, a system for efficiently processing geo-distributed big data. G-MR is a Hadoop based framework that can efficiently perform a sequence of MapReduce jobs on a geo-distributed dataset across multiple datacenters. G-MR acts much like the atmosphere surrounding the clouds. The problem of executing geo-distributed MapReduce job sequences as arising in "cloud-of clouds" scenarios is analyzed for job execution. For distributing the input data, DTG algorithm is applied to identify optimized execution path. The datacenter with optimized execution path is selected for job execution.we can implement PACT algorithm for improve the MAP reduce Schema. After implementing this we implement the Query Supporting System in the MAP REDUCE environment as followed by from the cloud mechanism. In this proposed approach we use query optimizing techniques to improve the results of the query as well as reduce the cost for the computation. To achieve this we use the indexing mechanism to the cache system to preserve the query search results.So for efficiency in the storage of the processing time in the case of big data we need to implement a system such as it will improve processing time and cost .As well known Map Reduce programming model is helpful for the handling geodistrtributed data sets, though it is not much efficient to handle complex jobs like weather forecasting, Population counting, stock exchange data etc. So for such data we are proposing a programming model which having the enhance structure of Map Reduce model.

Conclusion

From this paper it is concluded that PACT programming model is a generalization of MapReduce, providing additional second-order functions, and introducing output contracts. Although often more user functions need to be implemented, these have much easier functionality.The PACT programming model encourages a more modular programming style. Hence, interweaving of functionality which is common for MapReduce can be avoided. In this way we can say that over the extension of the Map reduce ,PACT programming model is more efficient for improving time and cost for the geodistributed data sets.

References

Alexander Alexandrov, Stephan Ewen, Max Heimel, Fabian Hueske,Odej Kao, Volker Markl, Erik Nijkamp, Daniel Warneke (2010), MapReduce and PACT - Comparing Data Parallel Programming Model, Technische University at Berlin, Germany.

ChamikaraJayalath, Julian Stephen, and Patrick Eugster(2013), From the Cloud to the Atmosphere: Running MapReduce across Data Centers, *IEEE transactions on computers*, Vol. 63, no. 1.

C. Olston, B. Reed, U. Srivastava, R. Kumar, and A. Tomkins(2013), Pig Latin: A Not-so-Foreign Language for Data Processing, *Proc. ACM SIGMOD Int'l Conf. Management of Data.* HadoopAcross Data-Centers.

Hadoop: The Definitive Guide, http://oreilly.com/catalog/ 9780596521981, 2013.

H. Chang, M. Kodialam, R. Kompella, T. Lakshman, M. Lee, and S. Mukherjee(2011), Scheduling in MapReduce-like Systems for Fast Completion Time, *Proc. IEEE Infocom.*

M. Zaharia, A. Konwinski, A.D. Joseph, R.H. Katz, and I. Stoica(2008), Improving MapReduce Performance in Heterogeneous Environments, *Proc. Eighth USENIX Conf. Operating Systems Design and Implementation (OSDI).*

M. Zaharia, A. Konwinski, A.D. Joseph, R.H. Katz, and I. Stoica(2008), Improving MapReduce Performance in Heterogeneous Environments, *Proc. Eighth USENIX Conf. Operating Systems Design and Implementation (OSDI).*

M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly(2007), Dryad:Distributed Data-Parallel Programs from Sequential Building Blocks, *Proc. ACM Second SIGOPS/EuroSys European Conf.Computer Systems (Eurosys)*

S. Agarwal, J. Dunagan, N. Jain, S. Saroiu, A. Wolman, and H. Bhogan(2010), Volley: Automated Data Placement for Geo-Distributed Cloud Services, *Proc. Seventh USENIX Conf. Networked Systems Design and Implementation (NSDI).*

T. Condie, N. Conway, P. Alvaro, J.Hellerstein, K. Elmeleegy, and R. Sears (2010), MapReduce Online, *Proc. Seventh USENIX Conf. Networked Systems Design and Implementation (NSDI).*

T. Condie, N. Conway, P. Alvaro, J.Hellerstein, K. Elmeleegy, and R. Sears(2010), MapReduce Online, *Proc. Seventh USENIX Conf. Networked Systems Design and Implementation (NSDI).*

V. Ramasubramanian, T. Rodeheffer, D. Terry, M. Walraed-Sullivan, T. Wobber, C. Marshall, and A. Vahdat(2009), Cimbiosys: A Platform for Content-Based Partial Replication, *Proc. Sixth USENIXSymp. Networked Systems Design and Implementation (NSDI).*

W. Lloyd, M.J. Freedman, M. Kaminsky, and D.G. Andersen (2011), Don't Settle for Eventual: Scalable Causal Consistency for Wide- Area Storage with COPS, *Proc. ACM 23rd Symp. Operating Systems Principles (SOSP).*