Indian Journal of Science and Technology, Vol 10(21), DOI: 10.17485/ijst/2017/v10i21/109290, June 2017
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

A Comparative Study of Big Data on Mobile Cloud Computing

Nikhil Rajyaguru and M. Vinay

Christ University, Bengaluru - 560029, Karnataka, India; nikhil.rajyaguru@cs.christuniversity.in, vinay.m@christuniversity.in

Abstract

Objective: To find the difference, understand and compare big data technologies which can be deployed to mobile devices. **Method/Analysis:** This research is a comparative study on recent trends and technologies that have taken place in past few years, which have been shifting towards mobile cloud computing. Technologies are compared on the basis of MapReduce Framework. The comparison is done with the help of results of certain test cases performed by the ongoing research in this field. Distributed computing is one of the common features among all the technologies. Use of graphs and framework design has been illustrated to identify the difference among these technologies. Framework design and its working have been briefly discussed to understand the flow of every technology discussed. **Findings:** The current technology consists of big data tools used only on computers. This research tries to see the benefit of same big data tools being used on mobile phones. With new tools and technologies coming to sort big data problems, this paper will help in identifying differences among them in detail which will help a user to decide what they should opt for. **Application/Improvements:** This research can be further improved with to keep track of upcoming versions of Hadoop or spark that will help in MCC.

Keywords: Big Data, Hadoop, MCC, MapReduce, Spark

1. Introduction

Technology has reached a stage where everything is at the user's fingertips. The phone isn't just a machine used to store information anymore. Mobile devices have evolved to a state where they are currently capable of competing with computers. They have a better memory for storage and computation with sensors at their disposal like the barometer, accelerometer etc. Recent trends have shown us growth in mobile traffic up to 63%.[1] With such a high growth among the mobile industry, research is being pushed to see how big data processing can be achieved among mobile phones.

1.1 Mobile Cloud Computing

MCC is a combination of Cloud Computing and Mobile technology with the help of the internet, which has progressed over time from reading emails and playing games to learning center on phones with help of applications, tracking a person's health and to entertainment [2]. Increase in the number of mobile phones and usage of internet by every user amounts to large datasets which can help companies to make products better and give benefits to different user as per needs.

Features like GPS, Accelerometer, Gyroscope, and Digital Compass are to be considered when we talk about MCC as these features can bring many applications to the table, but there are few challenges that exist in MCC which brings down the efficiency of at ask when deployed to it. Such disadvantages are as follows:

- Security: When everyone's data is stored on multiple servers around the globe, it is a big challenge for them to keep it secure.
- Battery: Once the battery is drained from mobile phones, all the features of MCC become void.
- Presentation: The information that can be portrayed on bigger screen compared to the screen on mobile phones are still limited due to its

*Author for correspondence

screen size, the accessibility to a graphical user interface such as viewing of an image and working on multiple screens.

- Cross Platform: With so many providers, the information that can be used to analyze is scattered due to business proposition, and the need to bring all the providers into one platform is a task that is difficult to achieve.

1.2 MapReduce

MapReduce is designed to process large amounts of data in parallel form among different computers that are stored traditionally in HDFS (Hadoop File System), which is broken down into smaller size and proceeded among nodes. Nodes are computing machines in a network which have specific functionality. For example, Hadoop Environment consists of DataNode and NameNode. DataNode is responsible for executing a task and storing data, NameNode is responsible for managing and assigning task in Hadoop Environment

MapReduce has 2 stages

- Map-Stage: It takes the files from Hadoop File System (HDFS) and processes the data. It is passed line by line to mapper function and broken down into smaller blocks.
- Reduce-Stage: Shuffle stage and Reduce stage are combined here together to process the given operation and then joined back and the result are stored in HDFS.

2. Study of Existing Research Work

2.1 Misco

In[2] discusses the application of MapReduce framework to mobile phones which supports the python framework. It focuses on how different application can gather data, and give results in real time via parallel computing. It describes the model where an application can run different tasks based on user request or can be automated. The model is represented by different applications running on a different phone which will be responsible for mapping and reducing individually. An application when gives a task to a node it can either be utilized for mapping or reducing.

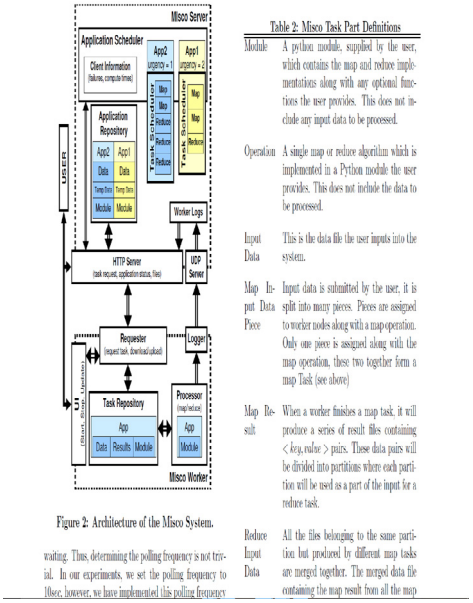

Figure 2: Architecture of the Misco System.

Figure 1. Misco Server.

Meta information doesn't reside on the worker node instead; the server keeps the information related to the task allotted to different nodes. Framework Design to communicate with the server and worker node Misco has adopted a poll-based approach. This approach utilizes information regarding a worker node is free or not by sending a poll information to the server, rather than the server keeping a constant watch over every worker node in the network. In Figure 1 the architecture of Misco Server shows in detail the working between different components.

The disadvantage of this approach is that polling frequency from worker node to server has to be adjusted where the worker nodes are under-utilized or over-utilized. If sent early and after that task arrives, the server has to wait for the next poll to be received for further task allocation to a worker node.

- **Data Transfer:** Communication and file sharing in Hadoop structure is done via Shared File System which is difficult to achieve in a mobile environment, for better result HTTP is used for the request, current task information etc. The Architecture consists of Misco Server, Misco Worker and Scheduler.
- **Misco Worker:** This is responsible for actual workload and execution of a task. It has 3 components which divide the work among them.
- Requester: Used as a communication bridge between server and worker via HTTP using URL

for requesting task, transfer of a file, starting the execution of files.

- Task Repository: For every task, data generated is stored in this component.
- Logger: Information of the currently running task.
- Misco Server: Responsible for allocation of task among workers and gather results.
- Application Repository: Keeps information regarding input data and output data among workers.
- HTTP Server: For communication among the workers, from which it receives the request for a task, completion of task etc.
- Scheduler: Which worker node should receive a task is decided by this component. Application Scheduler decides the order of task among application. Task Scheduler handles map and reduce from the application decided by application scheduler.

Implementation

Python is used in server and worker nodes, across the devices as it provides an easy development environment. Worker nodes use urllib module for downloading files to specific phones, while httplib is used for multi-request to the server on behalf of worker nodes.

Misco deals with different dynamic changes as well. With the help of upgrade of script among its worker nodes, it facilitates to give an overview and make changes when needed. When worker requests for a task to a server, it provides information via URL where worker node can download the needed files. If the server is free and has no task to assign, it will send a text saying "None" so that the worker will wait and be in an idle state.

Misco Server has parameters that it keeps while giving a task to worker nodes like a deadline, the data to be inputted, no of maps along with reduce partitions and optional parameters if required. Upon Job submission to a worker, the information is added to a directory with module, map and reduce partitions to view the progress of running task.

Analysis

Certain experiments were done to see the efficiency of this Framework and the results are shown in Figure 2. The majority of the time is spent on computation, while uploading, downloading and clean up (deleting of temporary files) is less time consuming. These results were achieved with the default scheduler present in the framework.

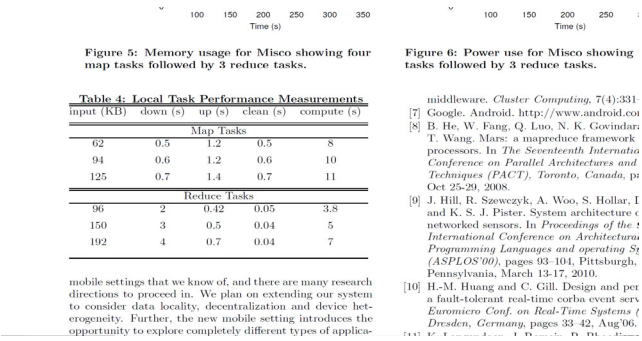

Figure 5: Memory usage for Misco showing four map tasks followed by 3 reduce tasks.

Figure 6: Power use for Misco showing tasks followed by 3 reduce tasks.

Table 4: Local Task Performance Measurements

input (KB)	down (s)	up (s)	clean (s)	compute (s)
Map Tasks				
62	0.5	1.2	0.5	8
94	0.6	1.2	0.6	10
125	0.7	1.4	0.7	11
Reduce Tasks				
96	2	0.42	0.05	3.8
150	3	0.5	0.04	5
192	4	0.7	0.04	7

mobile settings that we know of, and there are many research directions to proceed in. We plan on extending our system to consider data locality, decentralization and device heterogeneity. Further, the new mobile setting introduces the opportunity to explore completely different types of applica-

middleware. *Cluster Computing*, 7(4):331-
[7] Google. Android. http://www.android.co
[8] B. He, W. Fang, Q. Luo, N. K. Govindar
T. Wang. Mars: a mapreduce framework
processors. In *The Seventeenth Internatio*
Conference on Parallel Architectures and
Techniques (PACT), Toronto, Canada, p
Oct 25-29, 2008.
[9] J. Hill, R. Szewczyk, A. Woo, S. Hollar, I
and K. S. J. Pister. System architecture
networked sensors. In *Proceedings of the*
International Conference on Architectura
Programming Languages and operating S
(ASPLOS'00), pages 93–104, Pittsburgh,
Pennsylvania, March 13-17, 2010.
[10] H.-M. Huang and C. Gill. Design and per
a fault-tolerant real-time corba event ser
Euromicro Conf. on Real-Time Systems
Dresden, Germany, pages 33–42, Aug'06.

Figure 2. Local Task Performance Measurements.

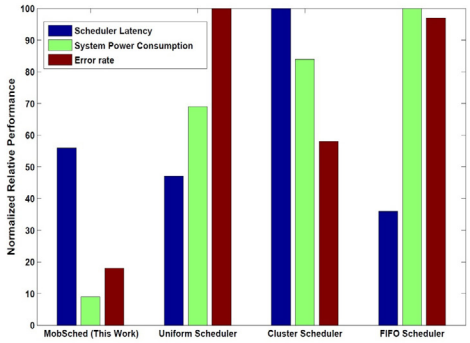

Figure 3. Comparison Different Scheduler against Mobsched.

Further research was done to find the difference among other schedulers. MobSched, a customizable Scheduler has been optimized for better performance on Misco[3]. In Figure 3, the difference between a traditional scheduler and optimized scheduler (MobSched) is significant.

The comparison in Figure 3 shows how efficient a scheduler is but fails to address certain issues, such as

- What happens when all there is a communication problem with a server and worker?
- Since communication among the nodes is taking place via HTML there is no method of securing material.

2.2 Hadoop MapReduce for Tactical Clouds[4]

This paper discusses about merging two technologies Hadoop MapReduce and MDFS (Mobile Distributed File System).

The paper establishes the theory suggesting traditional or proposed works are not secure like Misco and Hyrax. In hyrax, Datanode and TaskTracker of Hadoop environment were ported to android for distributed computing among various devices.[5] MDFS is similar to HDFS but has focused its implementation on mobile devices. MDFS was

initiated for military operations where all the data among soldiers could be collected and be secured in an event of loss of device and faster computation. These security measures are done by adapting k out of n approaches (k storage nodes, n-total number of nodes).

In Figure 4 it is shown how every block is encrypted with a secret key. The secret key is not stored in local database and for further secure measurements, the key is divided into parts using Shamir's Key sharing algorithm.

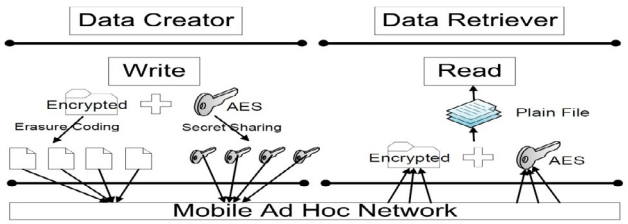

Figure 4. Mobile Ad Hoc Network for MDFS.

For accessing the data or a file stored in MDFS, the user needs at least k1 file fragments (< n1) and k2 key fragments (<n2).

Framework Design

MDFS Client: Operations to be performed on the MDFS are first initiated here; this is the interface between user applications and data stored in MDFS. File System takes arguments in URL format (scheme:/authorized:/path)

Data Processing Layer: Consists of Name Server and Data Server which have almost the same functionality like NameNode and Datanode in Hadoop. Name Server is responsible for storing namespace of MDFS. Data Server is daemon installed in MDFS client for job execution.

Network Communication Layer: It is responsible for communication among nodes in a network. It has 3 major components (Fragment Mapper, Communication Server, and Topology Discovery & Maintenance) which contribute differently to the task execution and carry blocks and messages among nodes.

Implementation

- MDFS clients
- Data server: MDFS daemon used for all the computation, initiated in every node.
- Name server: holds the namespace of a network.
- Fragment mapper: holds information regarding all the fragments and its location in the network.

Two different system architectural methods have been adopted in this research.

- Centralized System: This approach has a Master Server which looks over all the contents and activity taking place in the network. A user can invoke operations as mdfs client in a url format to connect with a network. This invokes Name server in a cluster whose task is similar to a namenode in Hadoop environment, holds information of data stored in hierarchy format with the authority given to every user at a different level. To access the information, user's access MDFS client that invokes the data server to give the information that is required with the help of name server information and the fragment server which tells which fragments are to be reconstructed with its key for authenticity to give access to the operation. The drawback of such a system is that failure of the main server leads to shutdown of the complete system.
- Decentralized System: All the above steps are performed independently (as every node acts as a master slave, having its own namespace and fragment mapping) when in operation. It collects information via communicating among all nodes in the network. The task is divided perfectly with the help of Job Tracker but the major flaw in this is the synchronization among all nodes and sharing of information. The network bandwidth is always high regardless of any real operation taking place as nodes will have to check for the number of copies on a regular basis. In a larger scale where thousands of nodes are in place, this will need a lot of computation and is time consuming if a large number of nodes are trying to perform operations over the network.

Operations

This paper has added new operations that can be performed on MDFS other than its 3 basic operations i.eread (), write(), list(). They have added append(), delete(), rename() and directory create(), delete(), rename() features are also given for more flexible and faster updating among namespace.

Analysis

Decentralized architecture is good when the number of nodes is less in the network, as it will perform its task quickly. It will become time consuming as it broadcasts information on a regular basis as it's difficult to implement on larger scale network with more than a thousand nodes. In a larger scale environment, it's easier to have

centralized architecture holding all the metadata for faster results. Figure 5 gives us an idea of how important communication among MDFS clients is as compared to Datanode in a traditional file system like HDFS. Due to minimum requirements to gather actual content from a block, a lot of bandwidth is utilized among the nodes alone.

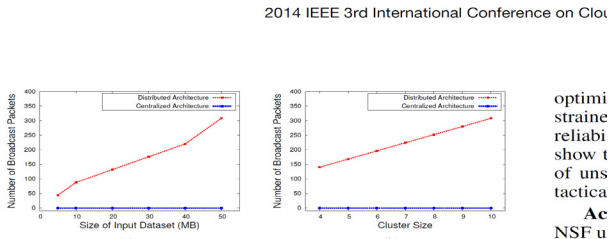

Fig. 10. Effect of (a) Cluster size (b) Input dataset size on Network bandwidth overhead in Centralized and Distributed Architecture

Figure 5. Performance Comparison of Distributed and Centralized Architecture.

2.3 Mobile Big Data Analytics using Deep Learning and Apache Spark[6]

This paper focuses on combining the task of data collection from various mobile devices and performs analytics on them. MBD (Mobile Big Data) is referred to be a growing opportunity for people to enhance the business aspects, along with healthcare and fraud detection.

Since parallel computing is a requirement in a majority of operations taking place in big data, the authors suggest how Spark technology can be used instead of Hadoop among mobile devices combining with Deep Learning. Comparison between Hadoop and Spark has shown Spark to have the upper hand and being faster by approximately 2.5x times[7].

Deep Learning: This is a branch of machine learning in which it learns from new inputs that are compared with old inputs of the user. It classifies every input into a different category based on an algorithm and later suggests its result for solutions of other similar options or problems faced by other users.

There are some open issues regarding this:

- Volume: With the increase in mobile data, how to handle a number of resources required to process the operations when it varies from time to time is an issue.
- Portability: Since every mobile phone isn't in a stationary location, the collection of samples from various devices becomes a major issue.
- Crowdsourcing: Due to many mobile users the collected data varies, due to this the quality of analytics or result produced in analysis cannot be assured, as it deals with different sets of preferences and usage depending on the its users.
- Training: Algorithms can learn the behavior based on some sample data or training data sets, but the data received from the devices are usually unlabeled and sample data are labeled sets of data.
- Time Utilization: The algorithm running on a mobile phone is slower compared to bigger machines which are dedicated for such purposes due to the volatile nature of MBD.

Learning patterns or deep models with mobile big data is slow in real time, as to rely on mobile resources for computing some algorithm which a standalone machine can do within a day compared to mobile which will take days is not reasonable. The paper therefore uses a technology called Spark which has shown improvement and growth in big data computation. Spark supports many features such as fault tolerance and recovery of operations.

Framework Design

Figure 6 is architecture proposed for deep learning using spark. The collected data from various devices are broken into portions which contain Resilient Distributed Datasets (RDD). Then the task of deep learning is operated on Spark. It consists of two main components:

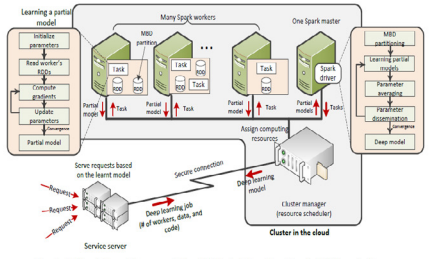

Fig. 3: A Spark-based framework for distributed deep learning in MBD analytics.

proposed framework helps in tackling the key characteristics of MBD.

V. Prototyping Context-Aware Activity Recognition System

Context-awareness [3], [5] has high impact on understand-

Figure 6. Spark based Framework for Deep Learning.

Spark Master: It initializes the task or operation given to it where the algorithm will be applied and it also decides which worker node will be assigned to each task

Table 1. Comparison among Proposed Framework

	Misco	**Hadoop MapReduce for Mobile Cloud**	**Spark on Mobile**
Security	Low	High	Low(Open Issue)
Centralized	Yes	Yes	Yes
Decentralized	No	Yes	No
Data Handling	Manual Data Collection	Manual Data Updating	Automatic Data Update
Data Analysis	Human Interaction	Human Interaction	Machine Analysis
Scheduling Algorithm	Sequential Task Scheduler	Energy Aware Scheduler	Not mentioned
Time Complexity (Task Scheduler)	O(N)	N^2logN	Not Mentioned

Spark Worker: Executes the task given by the master by taking the required information from RDD.

Implementation

Deep model in this system can be implemented in 2 steps:

- Gradient computation: It is the independent assessment of every dataset that is received by various devices in the network. (Map Stage)
- Parameter Update: In this, data models are created by combining the gradient computations and finding an average from all the data batches present in the network. (Reduce Stage)

The two steps mentioned above are used in Deep Learning in the following stages:

- Partitioning of Mobile Big Data: The complete data is split into partitions and stored in RDD, which helps in speeding up the process with the data residing in the main memory rather than in the disk.
- Deep Learning Parallelism: In this stage, MapReduce is performed on every data set to make it smaller and wait for the next iteration as input to compare and learn from it.

These steps are performed until the master server gets the desired result and a solution to its problem.

Analysis

Deep learning needs a heavier computational machine to give faster responses. The proposed architecture does not encounter or give a solution to veracity dimension of MBD. The use of spark technology is to minimize the computation required from various mobile phones to a data warehouse for mining purpose and analytics. The paper discusses about opportunities arise from the combination of deep learning and mobile cloud computing giving birth to mobile analytics. But the problem lies in the accuracy and results acquired from the algorithm implemented on deep learning. After resources utilization, if the analytics isn't accurate it's a waste of time and resource.

3. Conclusion

Mobile technology is now becoming a major part of every field[8] which includes health, online business, entertainment etc. Due to increase in Mobile Phones and their cheap availability to common people, there is a chance to analyze a lot of data. Table 1 summarizes all the information to be compared among the proposed framework discussed in the literature review.

All the above technologies have certain advantages over others but all of them work on Mobile Cloud computation which makes work portable compared to the traditional manner of working on the desktop. This paper focuses on trying to connect to a maximum number of small devices to distribute the workload and not be dependent on larger machines when an idle resource can be utilized around the world. But it also has some open issues that need to be resolved for better usage and adaptation. Few of the issues are:

- Same Service/ Cloud Provider: The information collected is not necessarily stored by one cloud provider, since there are many platforms for mobile. So collecting data on one platform to work on a solution is difficult.

- High Internet Speed: To implement this sort of regular monitoring of individual mobile devices it would require a lot of data to be transferred to the cloud and vice versa, which will require stronger internet speed at all time.
- Security: Due to rise in privacy matters, people don't like sharing information about themselves as they believe it will be used against them in the future if the information is acquired by an unknown party.

Future work can look into these issues to provide a better environment to implement proposed frameworks.

4. References

1. Mobile Traffic: Date accessed: 03/02/2016: Available from: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/mobile-white-paper-c11-520862.html
2. Dou A, Kalogeraki V, Gunopulos D, Mielikainen T, Tuulos V. Misco: A MapReduce Framework for Mobile Systems. PETRA '10 Proceedings of the 3rd International Conference on Pervasive Technologies Related to Assistive Environments. 2010. Crossref
3. Sindia S, Gao S, Black B, Lim A, Agrawal V, Agrawal P. Waco, TX, USA: Baylor University: MobSched: Customizable Scheduler for Mobile Cloud Computing 45th Southeastern Symposium on System Theory, 2013. 2013 March; p. 129-34.
4. George J, Chen C, Stoleru R, Xiey G, Sookoorz T, Bruno D. Hadoop MapReduce for Tactical Clouds 2014 IEEE 3rd International Conference on Cloud Networking. 2014; p. 340-46.
5. Marinelli EE. Carnegie Mellon University: Hyrax: Cloud Computing on Mobile Devices using MapReduce, Master Thesis. 2009.
6. Alsheikh MA, Niyato D, Lin S, Tan HP. Mobile big data analytics using deep learning and apache spark, IEEE Network. 2016; 30(3):22-29. Crossref
7. Shi J, Qiu Y, Minhas UF, Jiao L, Wang C, Reinwald B, Ozcan F. Clash of the titans: Mapreduce vs. spark for large scale data analytics. Proceedings of the VLDB Endowment. 2015; 8(13):2110-21. Crossref
8. Zhou B, Dastjerdi AV, Calheiros RN, Srirama SN, Buyya R.A context sensitive offloading scheme for mobile cloud computing service, Proceedings of IEEE International Conference Cloud Computing. 2015; p. 869-76. Crossref