RESEARCH ARTICLE OPEN ACCESS

Recognization of Satellite Images of Large Scale Data Based on Map- Reduce Framework

Vidya Jadhav, Harshada Nazirkar, Sneha Idekar, Prof. P.A. Bandgar.
Dept. of Information Technology,
JSPM's BSIOTR (W), Pune

Abstract
Today in the world of cloud and grid computing integration of data from heterogeneous databases is inevitable.This will become complex when size of the database is very large. M-R is a new framework specifically designed for processing huge datasets on distributed sources. Apache's Hadoop is an implementation of M-R.Currently Hadoop has been applied successfully for file based datasets.
This project proposes to utilize the parallel and distributed processing capability of Hadoop M-R for handling Images on large datasets.The presented methodology of land-cover recognition provides a scalable solution for automatic satellite imagery analysis, especially when GIS data is not readily available, or surface change may occur due to catastrophic events such as flooding, hurricane, and snow storm, etc.Here,we are using algorithms such as Image Differentiation,Image Duplication,Zoom-In,Gray-Scale.

Index Terms: Map-Reduce (M-R), HDFS(Hadoop Distributed File System) ,HIPI(Hadoop Image Processing Interface)

I. INTRODUCTION

Hadoop is a large-scale distributed batch processing infrastructure. While it can be used on a single machine, its true power lies in its ability to scale to hundreds or thousands of computers, each with several processor cores. Hadoop is also designed to efficiently distribute large amounts of work across a set of machines. Hadoop is built to process "web-scale" data on the order of hundreds of gigabytes to terabytes or petabytes. At this scale, it is likely that the input data set will not even fit on a single computer's hard drive, much less in memory. So Hadoop includes a distributed file system which breaks up input data and sends fractions of the original data to several machines in your cluster to hold. This results in the problem being processed in parallel using all of the machines in the cluster and computes output results as efficiently as possible.

The entire Earth surface has been documented with satellite imagery. The amount of data continues to grow as higher resolutions and temporal information become available. With this increasing amount of surface and temporal data, recognition, segmentation, and event detection in satellite images with a highly scalable system becomes more and more desirable. a semantic taxonomy is constructed for the land-cover classification of satellite images. Both the training and running of the classifiers are implemented in a distributed Hadoop computing platform.A scalable modeling system implemented in the Hadoop M-R framework is used for training the classifiers and performing subsequent image classification.

II. LITERATURE SURVEY

Performing large-scale computation is difficult. To work with this volume of data requires distributing parts of the problem to multiple machines to handle in parallel. Whenever multiple machines are used in cooperation with one another, the probability of failures rises. In a single-machine environment, failure is not something that program designers explicitly worry about very often: if the machine has crashed, then there is no way for the program to recover anyway.

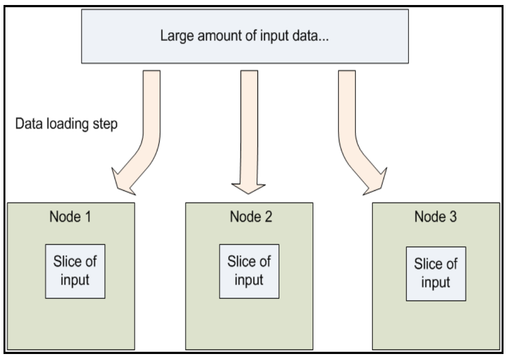
Performing computation on large volumes of data has been done before, usually in a distributed setting. What makes Hadoop unique is its simplified programming model which allows the user to quickly write and test distributed systems, and its efficient, automatic distribution of data and work across machines and in turn utilizing the underlying parallelism of the CPU cores.

Grid scheduling of computers can be done with existing systems such as Condor. But Condor does not automatically distribute data: a separate SAN must be managed in addition to the compute cluster. Furthermore, collaboration between multiple compute nodes must be managed with a communication system such as MPI. This programming model is challenging to work with and can lead to the introduction of subtle errors.

III. PROPOSED SYSTEM

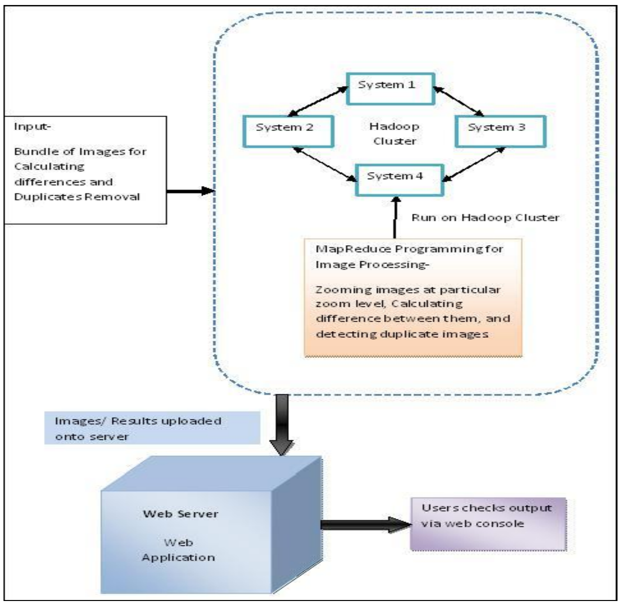
In a Hadoop cluster, data is distributed to all the nodes of the cluster as it is being loaded in. The HDFS will split large data files into chunks which are managed by different nodes in the cluster. In addition to this each chunk is replicated across several machines, so that a single machine failure does not result in any data being unavailable. An active monitoring system then re-replicates the data in response to system failures which can result in partial storage. Even though the file chunks are replicated and distributed across several machines, they form a single namespace, so their contents are universally accessible.

Data is conceptually **record-oriented** in the Hadoop programming framework. Individual input files are broken into lines or into other formats specific to the application logic. Each process running on a node in the cluster then processes a subset of these records. The Hadoop framework then schedules these processes in proximity to the location of data/records using knowledge from the distributed file system. Since files are spread across the distributed file system as chunks, each compute process running on a node operates on a subset of the data. Which data operated on by a node is chosen based on its locality to the node: most data is read from the local disk straight into the CPU, alleviating strain on network bandwidth and preventing unnecessary network transfers. This strategy of moving computation to the data, instead of moving the data to the computation allows Hadoop to achieve high data locality which in turn results in high performance.



IV. SYSTEM ARCHITECTURE

The system architecture includes the following-1.Large no. of images stored in file system.2.This Bundle of images is fed to hadoop distributed file system.3.On HDFS , we execute set of operations like duplicate image removal , zoom in and find differences among Images,using M-R Programs.4.The Result is then uploaded in web server,and shown to user through web application.


Input-
Bundle of Images for Calculating differences and Duplicates Removal
System 1
System 2
Hadoop Cluster
System 3
System 4
Run on Hadoop Cluster
MapReduce Programming for Image Processing-
Zooming images at particular zoom level, Calculating difference between them, and detecting duplicate images
Images/ Results uploaded onto server
Web Server
Web Application
Users checks output via web console

V. ALGORITHMS USED

1) Zoom in Algorithm

This algorithm takes the original image, creates four image tiles out of it, that means splits the original image into four pieces, re-draws each of the part on each of the four new images. Size of each of the new image is equal to the original size. For eg. Original image = 100*100 size, Split it into four parts of 50*50 size Each next we re-draw these parts into 100*100 size images. hence, we get four 100*100size images from original image of 100*100size.

2) Difference Algorithm

Here we divide the images into small chunks. We compare the respective chunks of image one and image two.

Comparison process :

1. Compare the intensity of the chunks
2. Compare the color codes of chunks

if chunks are different then mark the chunks with a red box.Again repeat the comparison process for all the chunks. And draw a new image with the red boxes marked i.e. showing the differences. Upload the difference image on tomcat server.

3)Duplication Algorithm

In this algorithm, we divide the images into small chunks. We compare the respective chunks of images. We have a sequence file with all the files of a binary data and it is the actual job that will filter & find the duplicates.

4)Grayscale Algorithm

A grayscale image is simply one in which the only colors are shades of gray. The reason for differentiating such images from any other sort of color image is that less information needs to be provided for each pixel. In fact a `gray' color is one in which the red, green and blue components all have equal intensity in RGB space, and so it is only necessary to specify a single intensity value for each pixel, as opposed to the three intensities needed to specify each pixel in a full color image. Often, the grayscale intensity is stored as an 8-bit integer giving 256 possible different shades of gray from black to white.

VI. TECHNOLOGY USED

M-R Framework

MapReduce is also a data processing model . Its greatest advantage is the easy scaling of data processing over multiple computing nodes. Under the MapReduce model, the data processing primitives are called *mappers* and *reducers* . Decomposing a data processing application into mappers and reducers is sometimes nontrivial. But, once you write an application in the MapReduce form, scaling the application to run over hundreds, thousands, or even tens of thousands of machines in a cluster is merely a confi guration change. This simple scalability is what has attracted many programmers to the MapReduce model.

- Pseudo-code for map and reduce functions for word counting

```
map(String filename, String document)
{
        List<String> T = tokenize(document);
         for each token in T
        {
         emit ((String)token, (Integer) 1);
         }
 }
reduce(String token, List<Integer> values)
{
        Integer sum = 0;
        Understanding MapReduce 13
         for each value in values
        {

        sum = sum + value;
        }    emit ((String)token, (Integer) sum);
}
```

The HIPI Framework

HIPI was created to empower researchers and present them with a capable tool that would enable research involving image processing and vision to be performed extremely easily. With the knowledge that HIPI would be used for researchers and as an educational tool, we designed HIPI with the following goals in mind.

1. Provide an open, extendible library for image processing and computer vision applications in a MapReduce framework
2. Store images efficiently for use in MapReduce applications
3. Allow for simple filtering of a set of images
4. Present users with an intuitive interface for image-based op- erations and hide the details of the MapReduce framework
5. HIPI will set up applications so that they are highly paral lelized and balanced so that users do not have to worry about such details

VII. ACKNOWLEDGEMENT

We take this opportunity to thank our project guide and Head of the Department Prof. P.A.Bandgar for their valuable guidance and mentoring through out this project. Additionally, we must give great thanks to Miss. Sneha Mam and Mr.Bhagat sir for guidance and support.

REFERENCES

[1] J. Dean and S. Ghemawat, Mapreduce: Simplified data processing on large clusters, Communications of the ACM, vol. 51, no. 1, pp. 107-113, Jan. 2008.

[2] T. White, Hadoop: The Definitive Guide, 2nd ed. O'Reilly Media / Yahoo Press, 2010.

[3] J. Talbot, R. M. Yoo, and C. Kozyrakis, Phoenix++: Modular mapreduce for shared-memory systems, in Proc. of the Second International Workshop on MapReduce and Its Applications, New York, NY, USA: ACM, 2011, pp. 9- 16.

[4] B. He, W. Fang, Q. Luo, N. K. Govindaraju, and T. Wang, Mars: A mapreduce framework on graphics processors, in Proc. of the 17th International Conference on Parallel Architectures and Compilation Techniques, New York, NY, USA: ACM, 2008, pp. 260-269.

[5] J. Ekanayake, H. Li, B. Zhang, T. Gunarathne, S.-H. Bae, J. Qiu, and G. Fox, Twister: A runtime for iterative mapreduce, in Proc. of the 19th ACM Int. Symposium on High Performance Distributed Computing, New York, NY, USA: ACM, 2010, pp. 810-818.

[6] ADAMS, A., JACOBS, D., DOLSON, J., TICO, M., PULLI, K., TALVALA, E., AJDIN, B., VAQUERO, D., LENSCH, H., AND HOROWITZ, M. 2010. The frankencamera: an experimental platform for computational photography. ACM SIGGRAPH 2010 papers, 1–12.

[7] APACHE, 2010. Hadoop mapreduce framework. http://hadoop.apache.org/mapreduce/.

[9] CONNER, J. 2009. Customizing input file formats for image processing in hadoop. Arizona State University. Online at: http://hpc. asu. edu/node/97.

[10] P. M. Atkinson and A. R. L. Tatnall. Neral networks in remote sensing. International Journal of Remote Sensing , 18(4):699–709, April 1997.

BIOGRAPHIES

V.D.Jadhavis pursuing B.E degree in Information Technology from JSPM's BSIOTR (W), University of Pune, Maharashtra, India.

H.J.Nazirkar is pursuing B.E degree in Information Technology from JSPM's BSIOTR (w), University of Pune, Maharashtra, India.

S.M.Idekar is pursuing B.E degree in Information Technology from JSPM's BSIOTR (W), University Of Pune, Maharashtra, India.

P.A.Bandgar, has received her Master's from SCOE, Pune, Currently working as a Head of the department in Information Technology atJSPM's BSIOTR (W), Pune, Maharashtra, India. Her Research interest is in the field of Image processing.