Advances in Computer Science and Information Technology (ACSIT)
Print ISSN: 2393-9907; Online ISSN: 2393-9915; Volume 2, Number 4; April-June, 2015 pp. 313-316

http://www.krishisanskriti.org/acsit.html

Satellite Image Processing with Hadoop

Aasim Ali[1], Anurag Singh[2] and Shashank Ramdokar[3]

[1]D.Y. Patil College of Engineering, Akurdi, Pune, INDIA
[2]Sapient Global Market, Gurgaon, INDIA
[3]SpadeWorx software services, Pune, INDIA
E-mail: [1]aasimali141290@gmail.com, [2]asingh212@sapient.com, [3]shashank.ramdokar@spadeworx.com

Abstract—*The entire Earth surface has been documented with satellite imagery. The amount of data continues to grow as higher resolutions of the images are available. With the rapid growth of multimedia data, it becomes increasingly important to develop semantic concept modeling approaches that are consistently effective, easily scalable and provide parallel processing for higher efficiency. Image processing algorithms related to remote sensing have been tested and utilized on the Hadoop MapReduce parallel platform it is also amenable to a broad variety of real-world tasks. We extend the file approach in Hadoop to regard the whole TIFF image file as a unit by expanding the file format that Hadoop uses. Finally, we apply this to other image formats such as the JPEG, BMP. Experiments have shown that the method is scalable and efficient in processing multiple large images and the difference in processing time in single pc and Hadoop is noticeable.*

Keywords: *Hadoop, Image Processing, MapReduce, HDFS*

1. INTRODUCTION

Satellite image data continues to grow and evolve as higher spatial and temporal resolutions become available. This study analyzes the recent advancements in distributed computing technologies as embodied in the MapReduce programming model and extends that for image processing of collected remote sensing images. The goal of Hadoop Image Processing is to create a tool that will make development of large-scale image processing and vision projects extremely accessible in hopes that it will empower researchers and students to create applications with ease. Our research has been conducted to find an efficient programming method for customized processing within the Hadoop MapReduce framework and to determine how this can be implemented. Performance tests for processing large archives of Land images were performed with Hadoop.

Many image processing and computer vision algorithms are applicable to large-scale data tasks. It is often desirable to run these algorithms on large data sets (e.g. larger than 1 TB) that are currently limited by the computational power of one computer. These tasks are typically performed on a distributed system by dividing the task across one or more of the following features: algorithm parameters, images.

The following image analysis algorithms were used: Difference Detection, Zoom, Gray Scaling, Duplicate Detection and Removal.

The findings demonstrate that MapReduce has a potential for processing large-scale remotely sensed images and solving more complex geospatial problems Current approaches to processing images depend on processing a small number of images having a sequential processing nature. These processing loads can almost fit on a single computer equipped with a relatively small memory. Still, we can observe that more disk space is needed to store the large-scale image repository that usually results from satellite-collected data. Format like GIF, BMP, TIFF, JPEG could be used on Hadoop image processing model.

2. HADOOP WORKING

Hadoop is Apache software that provides us with features of parallel processing and HDFS (Hadoop distributed file system). We can implement the parallel processing by writing program for Hadoop MapReduce function. HDFS is independent of MapReduce programming model. Hadoop HDFS provides us with fault tolerance, failure handling, file backup, scalability, and flexibility. HDFS contain master node and slave node for storage of large files. Large file is distributed among data nodes in pieces of 64MB-128MB and the size is optional and decided by the programmer. Master node (name node) contain the location of all the data that is been distributed among slave nodes (date nodes). The code that break the file into chuck (pieces) and merges output in MapReduce is hidden from the application user .The MapReduce library has two function map and reduces. The map jobs run on the entire data node in parallel and produce the output with sets of intermediate Key/value pair then it is shuffle before the output send to reduce function, it reduce the output in single set by merging the Key/values to produce appropriate output of the jobs. This output is read by appropriate application to display out for the query which is being fired by user.

3. ALGORITHMS

3.1. Difference Detection

- Make an empty image file of size height and width which calculated by comparing image 1 and image 2.
- Read each pixel first pixel is checked if it's the same as the pixel in the same spot in the second image then calculate Delta E (Euclidean Distance) for each.

$$\Delta E^*_{ab} = \sqrt{(L^*_2 - L^*_1)^2 + (a^*_2 - a^*_1)^2 + (b^*_2 - b^*_1)^2}$$

Delta E^*_{ab}: Euclidean Distance L^* scale: Light vs. dark where a low number (0-50) indicate dark and a high number (51-100) indicates light.

a^* scale: Red vs. green where a positive number indicates red and a negative number indicates green.

b^* scale: Yellow vs. blue where a positive number indicates yellow and a negative number indicates blue.

- If value of Delta E is >2 than show pixel with red in resultant image.
- Repeat till all pixels have been checked and resultant image is completed.

3.2. Duplicate Detection and Removal

There are two types of duplicate images those are Identical images and Similar images (different in Exposure or flash compensated, different focus point etc). Identical photo are easy to find and they can be found by matching their checksum or MD5.For finding the Similar images kind we have to follow these steps

- Divide the image into a 32 x 32 grid (1024 rectangles)
- For each of the rectangle in the grid, average the red, green & blue color channels and store in 3 separate 32 x 32 array.
- These 32 x 32 arrays represent the signature of the image.
- To compare two images, compute the signatures of both the images.
- Calculate the average of the corresponding array differences. i.e.

Similarity = 1 - (abs (red [image1] - red [image2]) + abs (blue[image1] - blue[image2]) + abs(green[image1] - green[image2])) / 255 * 1024 * 3

red: Red color of image

blue: Blue color of image

green: Green color of image

abs: Absolute

- is considered an exact match, while 0.0 for exact opposite images.
- >.95 is useful to find images that have been re-saved to other formats, dimensions, or compression.

3.3. Zoom In

- Make an empty image file with the double size of the original image.
- For instance, if the current size of the picture is 200X200, the new image size should be 400X400.
- Read each pixel of the image file and write the pixel in four neighborhoods of the new image file.
- Repeat step "third" for all pixels.

3.4. Zoom Out

- Make an empty image file with the half size of the original image.
- Read each pixel of the image file which is located in x and y, then write the pixel in x/2 and y/2 place.
- Repeat step " second" for all pixels.

3.5. Gray Scale

- Get the red(R), blue (B), green (G) value of pixel.
- Take the average of red(R),blue(B),green(G) i.e Gray= (R+G+B)/3 this is single gray value for a pixel.
- Replace the original red, green and blue values with new gray value.

3.5. Set Theory of Algorithms

- Let S be the Entire Set of images
- S = {I_1, I_2, I_3 …} where, I_i= are images in set of images for i= 1 to n
- Let D be the set of all the images which are duplicate of some or the other image in S
- D = {ID_1, ID_2 …} where ID_j = are images which are duplicates of some image or other image in S for j= 1 to n
- Let Z be the set of zoomed in Images
- Z = {IZ_1, IZ_2 ..} where IZ_k = four zoomed in parts of image I1 for k= 1 to n

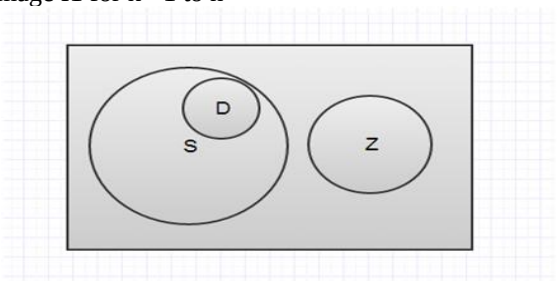

Fig. 1: Initial Image Set.

- Let F be the final Set of Images then F will be Final Set (F)of images contain Initial Set (S) + Zoomed in Images (Z) - Duplicate Images(D)

F = S – D + Z

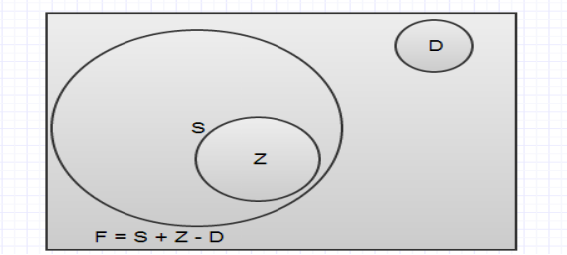

Fig. 2: Final Image Set.

4. SYSTEM IMPLEMENTATION

In this project we have made a web console to upload the image in Apache Tomcat and in background Oracle Database it is also keeping the records of user authentication, process status record running on Hadoop and also record of processes done on images on user request. First the use login with its authenticated id password and then select operation to be performed .The operation is fetched by job runner this runner make the sequence file of URLS of images and give it to Hadoop. Hadoop download the image and distribute it on HDFS. Operation are done on images with the help of MapReduce programs.

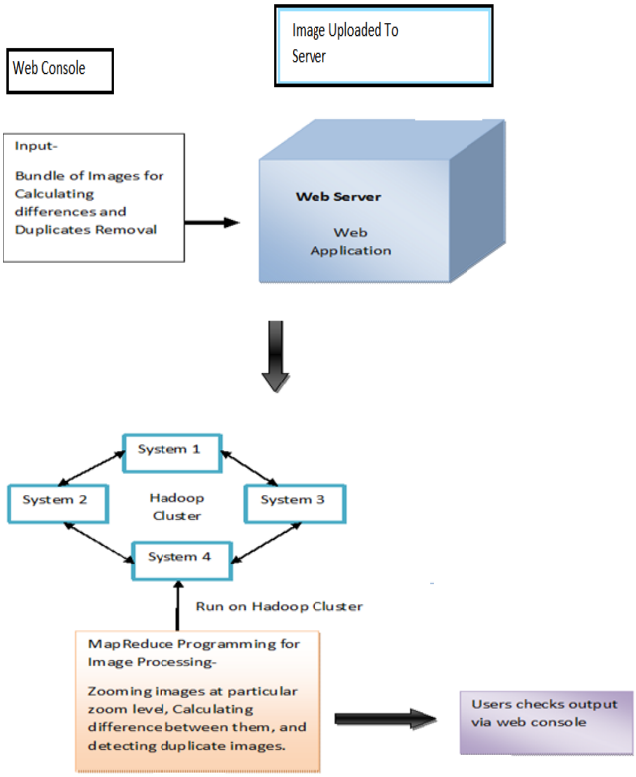

Fig. 3: System Model.

After the operation is done on image, sequence file is created and converted to appropriate format and uploaded on Apache Tomcat and the status is changes in oracle data base from queued to complete and user can download the image resultant image on to its PC. This is the flow of operation for all the operation except Zoom In and Zoom Out .Because we have written program in such a way that in Zoom Out and Zoom In operation the image is split in four parts so that Hadoop can do Zoom In or Zoom Out on different slave nodes individually and we get four resultant images and for all the images reducer make sequence file and converted to appropriate format like JPEG etc and resultant is uploaded to Apache Tomcat and user can download the resultant image directly from Apache Tomcat.

4.1. Environment Setup

We have setup Hadoop cluster with one Name node, two slave node and Personal Computer for keeping database .This Personal Computer act as client and it also have data base of operations .Beside this, all computation node have Ubuntu 12.04, Hadoop 0.20.1 and java 1.6 were installed on them . Table 1 show master slave configuration, Table 2 and Table 3 show cluster configuration.

Table 1: Master Slave Configuration.

Name	Number	Details
Slave node	2	2.5 GHz CPU, 4 Gb RAM,500Gb
Master node(Name node)	1	2.5 GHz CPU, 4 Gb RAM, 500Gb

Table 2: Cluster Hardware Configuration

Name	Number	Details
Slave node	2	2.5 GHz CPU, 4 Gb RAM
Master node(Name node)	1	2.5 GHz CPU, 4 Gb RAM
Storage	3	500Gb
Network		1 Gb Switch

Table 3: Cluster Software Configuration for Computers.

Name	Number	Details
Hadoop	0.20.1	Installed on each node
Ubuntu	12.04	Pre-configured with Java and Hadoop SDKs
Storage	3	500Gb
Java Net Beans		For Programming of MapReduce

Table 4: PC For Login.

Name	Number	Details
Personal Computer	1	2.5 GHz CPU, 4 Gb RAM,500Gb,preintalled oracle and Apache Tom cat,OS windows 7

5. CONCLUSION

In conclusion this paper presented the less time consumption, reliability, scaling of parallel image processing using Hadoop MapReduce frame work. Large no of images can be processed

parallel and operation can be done in parallel i.e processing algorithm can run in parallel to decrease time consumption of individual process .This project can be implemented on large cluster also, to process large data sets of images and suitable for different images format like JPEG,TIF etc. For companies which work with large image data sets our implementation is effective and scalable for them. In practice our algorithms will show good result when implemented on large cluster because as the cluster size increase the processing time decreases.

We have observed that Hadoop implementation is suited for large data sets .In future we can focus on different algorithms to be implemented on Hadoop which could provide higher efficiency.

REFERENCES

[1] Hadoop. http://hadoop.apache.org/.

[2] Noel C.F.Codella, Gang Hua, Apostol Natsev,John R. Smith. Large Scale Land-cover Recognition of Satellite Images. IBM T.J. Watson Research Center, Hawthorne, NY, USA.

[3] MapReduce. hadoop.apache.org/mapreduce/.

[4] Chris Sweeney, Liu Liu, Sean Arietta ,Jason Lawrence. Thesis on HIPI: Hadoop Image Processing Interface foe Image-based Map Reduce Task ,University of Virgina.

[5] Mohames H. Almeer. Hadoop MapReduce For Remote Sensing Image Analysis.ISSN 2250- 2459,Volume 2,Issue 4,April 2012.

[6] Algorithms. http://stackoverflow.com .