IJCTA, 9(12), 2016, pp. 5583-5589

Distributed Image Processing Using HIPI

***Dhinakaran K **Prasanthi K ***Pradeep Kumar**

Abstract : In today's world the collection of images are increasing which occupies huge memory. Such large collection of images cannot be analysed in our personal computer. So to process the images the distributed computing is needed. This paper describes the framework called Hadoop MapReduce Image Processing (HMIPr) framework. It provides an ability to implement the image processing technique in the distributed computing. Open source Apache Hadoop is used to implement the HMIPr. HMIPr is based on the map reducing technique. Various types of image representations are provided by the HMIPr in the internal format of Hadoop and input and output tools for the integration of image processing in to Hadoop data work flow. Common image processing libraries are used for the image format in Hadoop MapReduce Image Processing framework. It includes high level image processing API for developers not familiar with Hadoop. Sequential functions can be created using APIs to process one or group of related images. The HMIPr framework applies such function s to the large amount of images in parallel. HMIPr includes implementation of MapReduce for popular image processing algorithms, which is used for distributed image processing without the software development process. The proposed HMIPr Framework is more efficient compare to existing model.

Keywords : Bigdata, Image Processing, Map Reduce, HMIPr.

1. INTRODUCTION

Image processing is the frequently used technique in vast areas like medical image, astronomical data analysis and so on. The collection of images is increased to petabytes in last few years. Such a large collection of images cannot be analysed on a personal computer within a reasonable time. So that distributed computing is needed for image processing task.

Nowadays, various technologies of parallel and distributed computing are available. Commonly used technologies are MPI and MapReduce [7], but MapReduce is mostly used and preferred technology for image processing, because MapReduce technology provides automatic parallelization, fault tolerance and work will be distributed among the cluster nodes. So that the application developer can easily work on the image processing instead of dealing with the complicated distributed computing. In addition to that, distributed file system (DFS) is included in the commonly used MapReduce technology. This allows a large volume of storage of data on the commodity hardware. So that the storage cost of large collection of images are decreased significantly. The MapReduce technology will be implemented using the open source Apache Hadoop [3].

Image processing is mostly implemented using the Apache Hadoop although it is mainly intended for text processing. Hadoop doesn't support working with images out of the box. So that the developers should implement many routine tasks by themselves, which are common for image processing application based on the Hadoop. For example reading an image from the HDFS to memory, converting the image to the Hadoop internal representation and writing the image back to HDFS after processing [4].

* Assistant Professor UG Scholar Professor

** Department of Computer Science and Engineering, Rajalakshmi Institute of Technology

*** Saveetha School of Engineering, Birla Institute of technology Chennai, India, e-mail-maildhina.k@gmail.com1, prasanthi.karlapudi9 @gmail.com2,pradheep.kumar@pilani.bits-pilani.ac.in3

HMIPr framework, presented in the paper is aimed at the making of image processing in Hadoop easy and efficient to work on the distributed computing. The image representation provided by the HMIPr framework in the internal format of the Hadoop and the image processing input/output tools for integration into Hadoop data workflow[5]. Furthermore, the HMIPr offers the image processing API which will be useful for application developers those who are not familiar with Hadoop. The internal details of the Hadoop distributed computing environment will be hidden by the API from the developers and allows the applicationdeveloper to concentrate on creating the algorithm for image processing.

2. BACKGROUND

A. Hadoop Framework

MapReduce was developed by Google as a software frameworkfor parallel programming. It is mainly used to write an application which can process large amount of data in a parallel on large cluster of commodity hardware. The data are presented using list of key-value pairs. Processing of data consist of two computational phases called Map and Reduce and one communication phase shuffle and sort. The main advantage of the

MapReduce is ability to process different key- values pairs in parallel. MapReduce implementation provides automatic parallelization, so that the application developers need to crate only the serial map and reduce function.

Despite the restrictions of the MapReduce model, several image processing tasks will be easily represented as the MapReduce tasks. Each map function can process different related images, and reduce function combine the maps output to one resulting image, for example face recognition, in this case it is implemented using map-only job.

Fig. 1. Map Reduce Architecture for Hadoop Image Processing

Google introduce the MapReduce programming model but implementation is shared by it. Various open source MapReduce implementation were created, using the popular apache Hadoop. In addition Hadoop distributed file system MapReduce also included.

B. Hadoop Image Workflow

Image workflow is automated by the Hadoop in MapReduce (fig1) [10] [12]. From the data node the images are needed to be read first using the Input Format class in the Hadoop, it divides the input image into logical parts(image *j*, value *i*) called splits. The data node contains the n number of images, from the data node image and the value are taken and sent to the mapper. The mapper randomly selects the images and the values and it will we be processed in the separate mapper. Secondly the processed data will be grouped and sorted by image. The data will be stored in the local disk temporary storage. The reducer takes the sorted images and values grouped together and the values from value A to value N listed with the related image and that image will be the output image of the Hadoop image work flow.

3. RELATED WORK:

Image processing is implemented using Hadoop in several research areas. Hadoop runtime structure is complicated one so it is hard to extend, the standard Hadoop capabilities to process the image was tried to use by

the developers. The byte arrays to the image were converted from every map and reduce function, the images will be processed as required and the images are converted back to the byte array. This approach is not convenient but for most of the case it is efficient. There are different systems are available for implementing image processing in Hadoop: HIPI (Hadoop Image Processing Interface) and Opencv(open source computer vision) [1] [6] [11]. HIPI framework is used to enable the image processing in Hadoop. Opencv is the library for image and video analysis, Opencv tools can be implemented using Hadoop [8].The necessary parts of the Hadoop data work ûow for image processing will be provided by HIPI framework: the internal representation of image in Hadoop is based on float array of pixel.

Hadoop Writable interface for serialization of Float Image is used by HIPI[2]. In addition, small set of image processing operations, like colour conversion, resizing, cropping are included to the Float Image. The image should be packed into a HIPI Image Bundle before the image gets processed; this will be a nonstandard file format of HIPI, same as Hadoop sequence file [12] [13]. In addition image descriptors are included to the HIPI image bundles; it can be used in the bundle to filter the image and the index for quick image search. Performance of MapReduce image processing will be provided by HIPI image bundle is higher compared to standard Hadoop capabilities. The array of pixel will be provided to the application developer: algorithms for image processing will be implemented from the scratch. Lack of interoperability and poor functionality of HIPI are the main disadvantages. The images should be packed in to the HIPI image bundle so that it can't be read by other systems.

Open cv contains more image processing and analysis features than HIPI. Opencv needs packaging image ûles into one large ûle like HIPI. Standard Hadoop sequence files are used by Opencv for packing the image file into one large file. The images are represented as byte array in the Opencv. Hence the conversion of byte array to image is required for image processing and back in each function of map and reduce.

4. HMIPR FRAMEWORK

HMIPr framework is used for MapReduce image processing aimed to provide the simple and convenient image processing in Hadoop. Large amount of images are processed by the developers using the familiar tools in HMIPr, without knowing the details of distributed computing.

Based on the famous image processing libraries image representation is used by the HMIPr. So that the application developer can use the existing implementation of algorithm from libraries for image processing, instead of starting from scratch.

The internal details of the Hadoop distributed computing environment will be hidden the developers [14]. Simple interface for creating the serial framework will be provided by the HMIPr framework, which process single or group of related images. The image will be already stored in the memory in a convenient format. The HMIPR applies the serial function to the large volumes of images in parallel using Hadoop capabilities.

A. Architecture of the HMIPR framework

The architecture of HMIPr framework consist of 2 layers (fig2): MapReduce layer and HDFS layer. The MapReduce layer consists of two tracker job tracker and task tracker. The job tracker will monitor the task tracker and task tracker will allocate the job to the nodes. HDFS layer consist of data node and name node, the data node consist of data blocks which process the data from the name node and update the name node. The name node consists of backup node which will take care of data backup of the name node [15]; if any problem occurs in the name node the backup node will support the system. This architecture also consists of image processing libraries, image processing application program interface (API) and core component.

The core components layer gives the basic of image processing in Hadoop. This layer includes the image representation form in the Hadoop format which will be suitable for using it as values in MapReduce programs [9] and it provides the input/output tools to integrate image processing into Hadoop data workflow. The image processing API layer is used to hide the internal details of the Hadoop, MapReduce and distributed processing

from the application developer. Image processing API for developing the serial image processing function and the MapReduce driver for executing the function are included in this layer. The library layer MapReduce implementation of image processing algorithms used to create the distributed image processing without any software development.

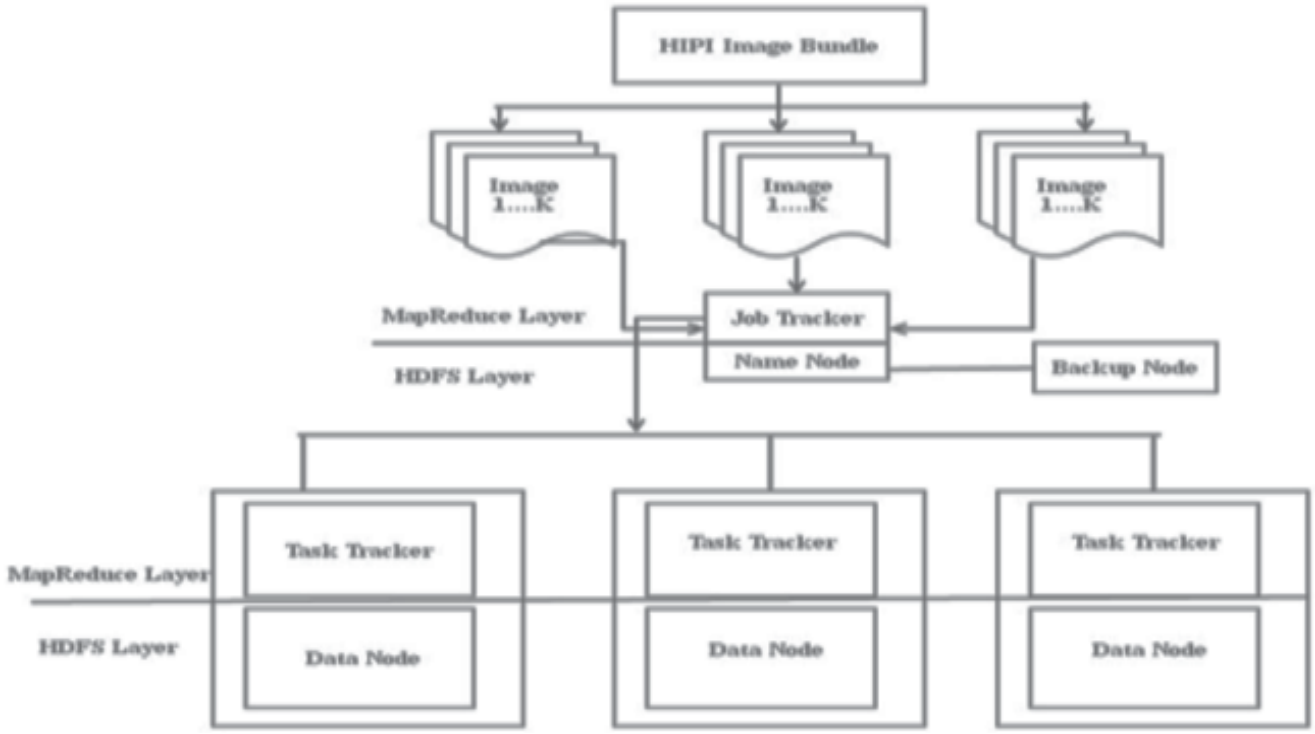

Fig. 2. Hadoop Distributed Image Processing Architecture

B. Internal Representation Of Image

The HMIPr framework consists of image representation formats based on the java 2D and Opencv. Serialization technology is provided using the Hadoop writable. Proposed format of the image is used as the values in MapReduce program. The keys in Hadoop must implement Writable Comparable interface instead of Writable so that images cannot be used as key. The keys in the Hadoop should implement writeable Comparable interface instead of writable.

C. Images Input/ Output Tools

For each type of representation of writable image special Input Format and Output Format were developed. To read the image form the HDFS Input Format was implemented, it create the desired wrapper writable image and a key-value pair of map input was produced. Null as the key and the image in the writable wrapper as a value is included in this pair. The entire image is process in this current implementation.

In traditional MapReduce approach the Output Format preserves the file name and extension of the original images. In representation of writeable image the images are stored as the metadata fields. Like HIPI and Opencv the HMIPr doesn't required packaging images into one large file. From HDFS the split can be read in one operation, it is faster than reading the small files separately. The advantage of Combine File Input Format is each mapper process different images instead of one. As a result the required mappers are declined and consequently, the overhead of starting and stopping of mappers are decreased. Combine File Input Format is an abstract class which requires the concrete implementations. Such implementations were created in HMIPr framework for every type of writable image representation.

The sequence files are used in the HMIPr framework for both input and output. Tools for packaging image datasets in to the sequence files and for extracting images for extracting images from such files were included in the HMIPr.

The processed images were written again in to HDFS as a single image using the Output Format.

Table 1. Operations Usined in HMIPr

Operation	*HMIPr image representation*	*library*
Conversion from JPEG to PNG	Buffered Image Writable	Java 2D
Edge detection	Buffered image writable	Java 2D
segmentation	Image writable	Opencv
Face detection	Image writable	Opencv

D. Image Processing API

The current HMIPr implementation includes a java image processing API. Main class of the API is an image processor with one method called process image. The source image is received form method in the necessary format as an argument and must return the processed image.

The image processing API contains several MapReduce drivers to execute the functions of image processing; application developer provides it as a Hadoop job. The appropriate image representation format Input Format and Output Format was stepped by the MapReduce driver. After that, the job on the Hadoop cluster is executed by drivers. Current HMIPr implementation provides Map-only drivers for Hadoop jobs. The mappers are used to process the image and each mapper deals with single image. The process Image method of the image processor class is called to process the image. The processed images are generated form the mapper and written in the HDFS

E. Algorithm Of HMIPR

INPUT : Image Bundle

OUTPUT : Segmented Image

STEP 1 : Randomly Sampled Image Bundles

STEP 2 : compute the average (mean) of n number of random image

STEP 3 : Computation of the covariance matrix in the same sampled images

STEP 4 :convariance Matrix

- Covariance float images to gray scale images
- Covariance Map Reduce
- Covariance Reduce process

(*i.e.*) INPUT : Image_j, value_i

Map : Image A,value A.....Image *n*,value *n*

Reducer : ImageA , list(Value *a*,Value *b*.....value *n*) Image B,list(Value *a*,Value *b*..... .value *n*)

5. EXPERIMENT

An experiment was continuously carried out to evaluate the scalability of HMIPr framework and performance is compared with HIPI and Opencv. The collections of images were used as a dataset in the experiment. The experiment was performed on the 3-node Hadoop cluster. The cluster consist one management node and two computing node with the following configuration: Operating system- linux (ubantu 14.04), cpu- intel i7 processor, ram-8gb, Hadoop 2.7.1.

A. Scalability

To evaluate the scalability of HMIPr the four image processing operation (image conversion, edge detection, segmentation, face detection) were performed on the image dataset. The image dataset will be packed into the sequence files before processing. The experiment was conducted in two series. The purpose of first experiment series is to evaluate how the HMIPr frame work scales with the increasing number of nodes in the cluster. The entire image dataset was processed on the Hadoop cluster using several numbers of nodes. The result is shown the figure. The next series of experiment was aimed to estimate the scalability of the HMIPr with increasing the image volumes. During the second series of experiment the entire Hadoop cluster was used to process several numbers of images. The second series of experiment result are shown in the fig..

As shown in the fig.3 the HMIPr framework has near-linear scalability both with the increasing the number of nodes in the cluster and increasing the number of images to process. the images are processed independently this shows the scalability of the HMIPr framework.

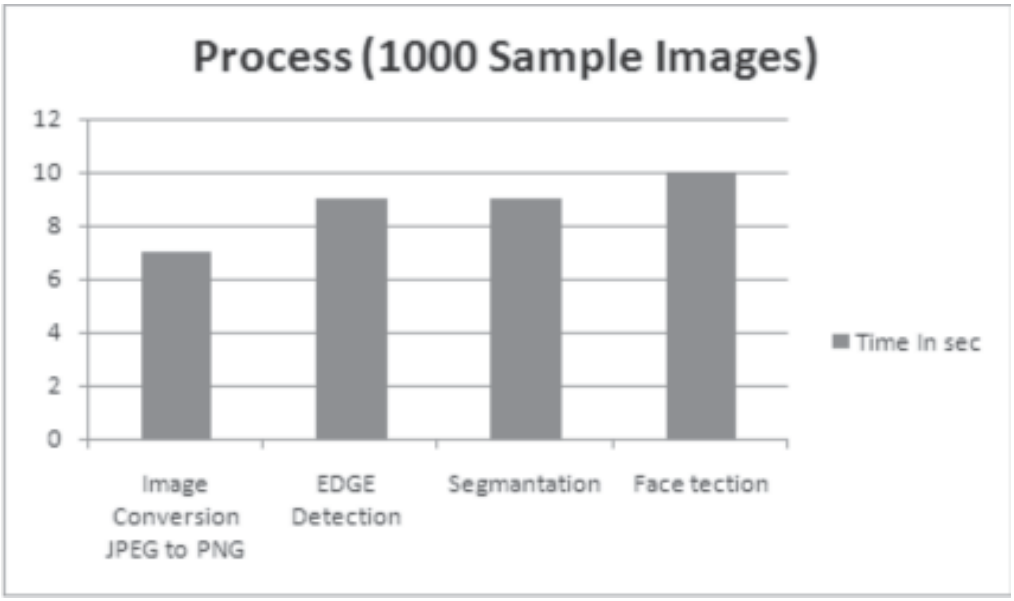

Fig. 3. Image Processing time for sample images

6. CONCLUSION AND FUTURE WORK

The HMIPR framework for distributed image processing using Hadoop is presented in this paper. The framework extends the Hadoop by providing the capability of using images in the MapReduce programs. The images in the HMIPr framework are represented based on the image processing libraries (java 2D and Opencv). By using this library the performance is increased and the processing time is been reduced. Scalability and performance of the HMIPr framework is demonstrated. In contrast to HIPI and Opencv the HMIPr framework is able to process images not only in large file but also the performance of small file processing is improved with the help of Combine File Input Format.

In future video will be analyzed and the object will be detected from the video file as an image , So that we can use this method in surveillance camera for detecting the object from the video file which is captured from the camera and we can use it for security purpose.

7. REFERENCE

1. Alberto M. C. Souzaa, Jos´e R. A. Amazonasb," An Outlier Detect Algorithm using Big Data Processing and Internet of Things Architecture", Procedia Computer Science 52(2015).
2. AndreySozykin and TimofeiEpanchintsev," MIPr – a Framework for Distributed Image Processing Using Hadoop", Application of Information and Communication Technologies (AICT), 2015 9th International Conference on 14-16 Oct. 2015,IEEE.

3. Apache hadoop. [Online]. Available: https://hadoop.apache.org. [4] Dhinakaran K, Silviya Nancy J and Saranya R," Classification and Prediction of Earthquake and Tsunami using Big Data Analytics", Australian Journal of Basic and Applied Sciences, 9(11) May 2015, Pages: 611-616.
5. Dhinakaran K., Silviya Nancy J. and Duraimurugan N," Video Analytics Using HDVFS In Cloud Environment", ARPN Journal of Engineering and Applied Sciences, VOL. 10, NO. 13, JULY 2015
6. Chin-Ho Lin, Liang-Cheng Huang, Chih-Ho Liu, Han-Fang Cheng, I-Jen Chiang," Temporal Event Tracing on Big Healthcare Data Analytics", 2014 IEEE International Congress on Big Data
7. J. Dean, S. Ghemawat, "MapReduce: simplified data processing on large clusters," Commun. ACM, vol. 51, no. 1, pp. 107–113, Jan.2008.
8. HIPI: http://hipi.cs.virginia.edu/
9. Katarina Grolinger ," Challenges for MapReduce in Big Data", Electrical and Computer Engineering Publications,2014.
10. Mehdi Bahrami, "Cloud Template, a Big Data Solution", Journal of Soft Computing and Software Engineering , Vol. 3, No. 2, pp.13-17, 2013.
11. OpenCv: http://opencv.org/
12. Sangwhan Cha and Monica Wachowicz," Developing a Real- Time Data Analytics Framework using Hadoop",2015 IEEE International Congress on Big Data.
13. Tan, Wei, et al. "Social-Network-Sourced Big Data Analytics" Internet Computing, IEEE 17.5 (2013): 62-69.
14. Mehdi Bahrami1 and MukeshSinghal ,"The Role of Cloud Computing Architecture in Big Data", Springer International Publishing Switzerland 2015.
15. W. Zeng, Y. Yang and B. Luo, "Access control for Big Data using data content," IEEE International Conference on Big Data, 2013.