Volume 6, Issue 5, May 2016 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Efficient Image Recovery for Enhancement Using Hadoop MapReduce Techniques

M. Anand, V. Mathivanan
Research Scholar, Amet University, Kanathur, Chennai,
Tamilnadu, India

Abstract: ***The mass development of social media network and increased computational power, solving most of the resource concentrated problems like Deoxyribonucleic acid (DNA) analysis could to able to solved with minimum effort. On the other hand, the quantity of image being uploaded in the Internet sites is increasing rapidly. Example: Facebook users uploading over 2.5 billion new photos every month [1]. Since the images are being shared in different ways, people start compressing the images to reduce the huge amount of memory space. This compression leads to data loss (pixel) in images which affects the quality of the images. Most of the recent computer applications use a small number of input images because of the difficulty in acquiring resources and storage options for processing large amount of data to enhance the quality of the image. For solving processing tasks involving data of this quantity, there are many MapReduce model of distributed processing of which Apache Hadoop is the most prevalent open source employment. It is know that using MapReduce model is a good solution for many problems; it is unknown that how the suitability of this model will work with regards to image processing. The work explains here explores the feasibility of using this technology for enhancing the image through recovery process.***

Key words: Image Enhancement, Hadoop andMapReduce.

I. INTRODUCTION

Due to the latest improvements in digital camera performance, the range of low-priced digital cameras and the digital camera equipped with mobile phones a large amount of image data is generated, and stored in some form of database. At this time, there are limits on what can be done to improve theperformance of single computers to make them able to process large-scale information, such as in Image processing toimprove the quality of the image. So the need for analysis techniques to take benefit of useful data that can be extracted from such processing is required.

Unfortunately there are limits on what can be done to improve the performance of single computers to make them able to process large-scale information. Hence, the advantages of parallel distributed processing of image databases by using the computational resources of a computing environment should be considered. Hadoop[2], provides really promising mechanism for processing large numbers of databases through parallel and distributed computing. In fact, Hadoop is usedby many companies, researches all over the world[3]. Readings using Hadoop have been made to treat one text data file or multiple files as a single file unit, examples like the analysis of large volumes of DNA sequence data, converting the data of a large number of images to any document format, and carrying out feature selection/extraction in astronomy[4].

Hadoop distributed file system (HDFS) [5] is developed as an open-source project tomanage storage and parallel processing of large scale data.HDFS parallel processing structure is based on MapReduce [6], [7], [8] programmingmodel that is introduced firstly by Google File System (GFS) [9] in 2004. MapReduce is aframework for processing highly distributable problems across huge databases using many numberof computers (nodes), collectively referred to as a cluster.

II. PROBLEM IDENTIFICATIONAND RELATED WORK

The image processing performed currently runs through ordinary sequential ways to accomplish the job. The Algorithm or program loads images one by one and process each image alone before writing the newly processed image on a storage device. Frequently people use very ordinary tools that can be found in Microsoft Photo viewer or Microsoft Paint and at the extreme case Photoshop, for example. Besides, many ordinary C, C++ and Java programs can be downloaded from the Internet or easily written to perform such image processing tasks. Most of these tools run on a single computer with a Windows operating system. Although groupor collective processing can be found in these single-processor programs, there will be problems with the processing due to limited abilities. Therefore, the need of a new parallel approach to work effectively on massed image data is mostly appreciated.

HDFS is expert in storing and processing files with large size. Usually storage andprocessing offiles with smallsize ends up withperformance decrease in HDFS. The file systemmanagerin HDFS is known as NameNode, and all the information related to the files are registered as metadata in themaster node. When using substantialamountof small size files, the memory requirement of NameNode increases whichleadsthe master node to be not responsivefor file operation

requests from client nodes [10]. Moreover, number of tasks to processthese files increases.Hence the operations initialize and execute tasks using TaskTrackers and JobTracker mechanismin Hadoop,have more tasks to schedule. In that way, overall Hadoop Distributed File System job execution performance decreases. Forthese reasons, processing and storing hugecollections of images require different techniques inHadoop.

Dong et al. propose two techniques for this problem [10] [11].In [10]heproposed file merging and perfecting scheme for structurally related small files, and then files grouping and perfecting for logically related small files. This approach is based oncategorization of files based on their structural and logical properties. In [11], he proposed anothersimilar approach on the same problem but introduced a two-level perfecting mechanism toimprove the efficiency of accessing small size files. He also usedto represent usecase situation with the help of Microsoft power point files.

Many researchers use Hadoop in imageprocessing. Most of them implemented Hadoopfor content-based image searching. Krishnaet al. [12] offers a Hadoop file system for storage and MapReduce standard for processingimages sneaked from the web. The input to the HDFS is a list of image URLs and thecontextual information also called the metadata of the image. Searching is done through the help of the metadata of the images. The only critical factor in such application is providing the key and value pairs and defining map and reduce functions respectively.

III. HADOOP- A DISTRIBUTED FRAMEWORK

Hadoop is an opensource structure for storing, processing, and analysis of large amounts of distributed and unstructured data [5]. The main origin of this processing framework returns to Internet search companies, Yahoo and Google. These companies need new processing tools and models used for web page indexing and searching.

HDFS framework is designed for parallel data processing at Petabyte and Exabyte scales which are distributed on the normal computer nodes, in such a way that a Hadoop cluster can easily be extended in parallel fashion. Hadoop is currently developed and expanded by The Apache Foundation.

Doug Cutting and Mike Cafarella in 2005 created the Hadoop framework. Cutting, who was working for Yahoo at the time of developing HDFS, named it after his son's toy elephant. It was formerly developed to support distribution for the Nutch search engine project [5].

The Apache Hadoop framework consists of 4 major modules which have been described below [5]:

- Hadoop Common – a general module which contains utilities and libraries needed by other Hadoop modules.
- Hadoop Distributed File System (HDFS) — it is a filesystem that stores data on service machines using distributed mechanism by providing hugecollective bandwidth throughout the cluster.
- Hadoop YARN—a platform used for managing resource in the HDFS and is responsible for managing resources allocation in each clusters and using them for scheduling of users' applications. This part also named MapReduce 2.0 (MRv2).
- Hadoop MapReduce—a model used for huge amount of data processing with the help of programing code.

3.1 Hadoop Distributed File System

HDFS is an open source application of the Google file system (GFS). Though it appears as ausual file system, the main purpose of it is toprovide a fault-tolerant storage structurecapable of holding huge amountsof data. It also allowsvery fast accessto the said data and it provides a way for MapReducemechanism to performcomputations on the same location where the data resides. HDFS is builton master-slave architecture principle (figure 3.1). The masternode known as name node provides data service and access permissionto the slave nodes, while the slave node also known as data nodes work asa storage device for the HDFS. Huge sized files are distributed and divided further among different data nodes. In each node the map processing jobs will be performed on their local copies of the data. It can be observed that the name node stores only the log information and the metadata while the data transfer to and from the HDFS is done through the Hadoop API.

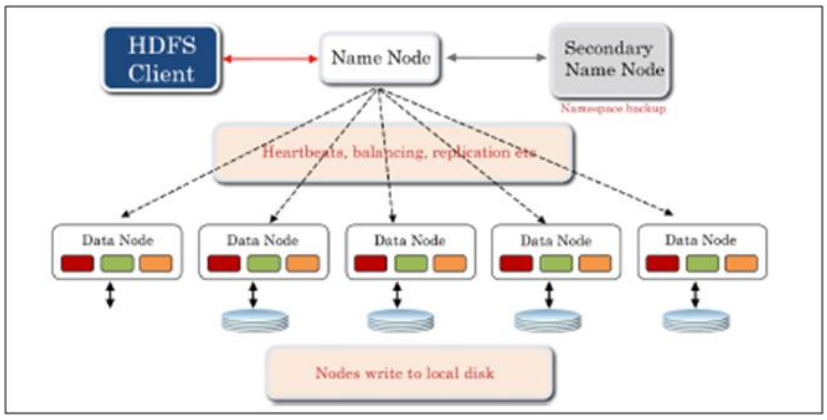

Figure: 3.1 Structure of HDFS file system

3.2 Map Reduce

MapReduce is an efficient programming model for processing parallel large amounts of metadata on cluster computers with weak and unreliable communication links. MapReduce mechanism is created based on the scaleout standard, which involves grouping a large number of desktop computers. The main working principle of using

MapReduce is to transfer computations to data nodes, rather than bringing data to computation nodes, and thus fully utilize the advantage of data locality feature. The code that divides work, exerts control, and merges output in MapReduce is entirely hidden from the application user inside the framework.

As long as shared and synchronized global states are not required most of the parallel applications can be implemented using MapReduce. Computation in MapReduce will be done in two stages: first the map stage and second the reduce stage. The data are represented by split sets of keyvalue pairs and their instances are managed in parallel by the map stage, the slaves are assigned by parallel number of instances that matches the node number.

This process generates temporary intermediate keyvalue pairs that can later be used to reduce stages. The processing is conducted in parallel within map stages or reduced stages. The reduce and map stages occur in a successive manner by which the reduce stage starts after the map stages finishes.

3.3 Merits and Demerits of Hadoop

One of the most important advantages of Hadoop is the ability to analyze and process large amounts of formless or partially structured data which have been impossible to process efficientlywith minimum cost and time so far [13].

The second advantage of Hadoop is that it can be extended with simple steps and its nature of horizontal scalability. The image data that need processing can easily be analyzed up to Exabyte level and there is no need for worries to work on example data and a subclass of the original data. With the help of Hadoop, the possibility of checking all types of data is provided.

Since it is freely distributed, the next advantage is its low set up cost and there is no need for the researches to buy professional and expensive hardware. In particular, it takes only a few hours to set up a Hadoop system due to the spread of cloud computing and its reasonable prices for case processing of data as well as private clouds[14].

The important point to note here is Hadoop and its subsets are all in the early stages of development and they are immature and unstable. This will lead to permanent modification of this framework that imposes costs of continuous training on organizations.

Due to the uniqueness of this HDFS software model, very few people have the required skills for establishing and working on Hadoopbased file systems. Shortage of expert people in Hadoop is the most important challenge of many concerns in using this system.

Evaluating different algorithms in this areacauses the lack of valid standards and benchmarksdue to the novelty of this technology. One of the few efforts in this area was attempted by Bajcsy *et al*. to assess four different methods of Hadoopbased image processing on cluster [15]. Hence we are still far from establishing inclusivestandards which are acceptable to academic community.

The next important issue in Hadoop's processing is,due to the inherent nature it is lack of the processing ability to real-time data. The request tracker in HDFS framework must wait for each compute node in the system to finish the work, and then it willdeliver the final answer to the user. However, this problem will be solved to some extent by the hasty growth of database technologies that don't support SQL and its combination with Hadoop. Moreover, frameworks such as Storm [16] and Samza [17] can also be used for real-time processing of high volume data.

IV. IMAGE ENHANCEMENT USING HDFS

There are many enhancement techniques have been discovered under global and local image enhancement techniques. The effectiveness of various methods has emerged out from the respective output image. The results obtained are sufficient enough to prove the effectiveness and usefulness of all these techniques in image enhancement field.

There exist a variety of spatial and frequency domain methods that achieve the same.Few of Enhancement techniques are:

a. Contrast Stretching [18]
b. Histogram Equalization [19]
c. Spline Approach [20]

The result obtained by this technique will result better for some images (Figure 4.1 and 4.2) where the adaptive color model (RGB) ratio is same throughout the images. It will not give the best for other types of images where the RGB is scattered throughout the images. For example while enhancing the image of capsicum shown below will give better result than the image of feathers.

More over the current processing of images goes through general sequential ways to accomplish this job. The program heaps image after image, treating each image alone before writing the newly processed image on a storage device. Generally, many people use very simple tools that can be found in Adobe Photoshop, for example. Besides, many usual C and Java programs can be downloaded from the internet or easily developed to perform such image processing tasks. Most of these tools run on a single computer which runs on Windows operating system. Although batch processing can be found in these single-processor programs, there will be problems with the processing due to limited capabilities. Therefore, there is a need of a new parallel approach to work effectively on massed image data in-order to get the efficient results.

4.1 Implementation

The main aim is to simply use each map job for reading an image file, process it, and finally store the processed image by sending into an HDFS assigned memory area. The name of the image file is considered to be the Key and its byte content is considered as the Value. Here, though the Key and the Value for the input parameters of the Map job

remains the same, we still use the output Key value completely. This Key value will be assigned to the name of the corrupted image file which needs enhancement. Reducers in this exception will collect all the file names that have undergone detection and will finally be collectedas a single output file stored in the HDFS assigned memory location.

The MapReduce framework is essentially designed for text processing and hence some changes in Java classes are inevitable for image processing needs. In order to customize Hadoop for performing image processing functions there are few important things to be considered, firstly all should have an idea of how to adapt the Hadoop data types, classes, and functions to access (read and write) and process binary image files which can be seen similarly in serial Java programs running on a computer. This can be performed by converting the Hadoop API from TextWritable class to BytesWritable class which can perform the required function on the image file and stop the splitting of input image files into chunks of many data files.

It is also important for us to remember that one can use MatLab for processing the images but as expected images do not correlate perfectly because we are usingdistanttype of inputs to our experiments, and our display of positiveand negative values of pixels also may differ slightly. Performing analysis on a massive image set needs unrestricted parallel knowledge. HIPI excels in these.

In order to process massive images the image is needed first. Following is the simple steps explain how to collect the large set of images:

Step 1: Specify the set of images to be collected.
Step 2: Group the images (URL) based on number of nodes and send assign each group to its respective mapper.
Step 3: Download images from the internet using JAVA URL connection class.
Step 4: Store images in a HIPI image format for Enhancement process.

Now then after collecting the required images as described above, the next step is to obtain the required pixel information from all the images. This can be performed by many clustering algorithms. The algorithm used here is given below:

Step 1: Load the image with size Height (H) x Width (W) in each clustered node.
Step 2: Calculate the total pixel (TP) value using TP = H * W
Step 3: Identify the required pixel (RP) value of the x, y coordinate using RP = x+(y*W)
Step 4: Extract the RGB value of the pixel using
float r = red (img.pixels(RP)), likewise for g and b.
Step 5: Updated the extracted pixel value in the original image to be enhanced.
Repeat the above steps in all the clustered nodes.

4.2 Pixel Sharpening

Sharpening is one of the most frequently used alterations that can be applied to an image for enhancement purposes, and there are many ways to do sharpening techniques but it is possible to use the ordinary methods of sharpening to bring out the image details that were not deceiving before. Image sharpening is used to enhance both the edge and the intensity of theimage in order to obtain the apparent image.

The sharpening method is implemented using convolution, which is an operation that calculates the required pixel by comparing the source pixel and its neighbors by a convolution kernel using the formulae ①. The kernel is anundeviating operator that explains how a stated pixel and its neighboring pixels affect the value computed in the destination pixel of the image due to a filtering operation. Specifically, the kernel used here is represented by matrixes with dimensions 4x4 throughdecimalpoint numeric value. When the convolution operation is performed, this 4 × 4 matrix is used as a sliding mask to operate on the pixels of the source image. To calculate the result of the convolution for a pixel located at x and y coordinates in the source image, the center of the kernel is positioned with respective to the coordinates. To calculate the value of the destination pixel at x and y coordinates, a comparison is performed on the kernel values with their equivalent color values in the source image. The source image is then be updated with the following over operation performed on it:

$$P_o = \frac{P_a \propto_a + P_b \propto_b (1 - \propto_a)}{\propto_a + \propto_b (1 - \propto_a)} \qquad ----- ①$$

Where,

P_o is the resultant pixel value of the destination image after performing over operation with source and destination image

$\propto_a$, $\propto_b$ are the alpha of pixels in source and destination image.

4.3 Reducers

Translatingmany smallsize files into one large size file is necessary to decrease the number of tasks and then process technique is implemented on this single large file. For merging many smallsize files Hadoop supports SequenceFile mechanism [21]. To solve the small fileproblem in HDFS SequenceFile is the most common solution. Many small files are gathered as a single largesize filecontaining small sizefiles as indexed elements in <key, value> format. File index information is the Key and thefile data is the value. The conversion is done by performing a conversion job that gets smallfiles as input value andproduce SequenceFile as output. Although general performance is increased with SequenceFile usage,it's very much important to make note that the image formats of the input images will not be preserved their after merging. For each addition of new input image setPreprocessing is also required. SequenceFile cannot directly access the SequenceFile and hencewhole SequenceFile has to be processed to obtain an image data as single element [22].

Next, combining set of images as one InputSplit technique is implemented to optimize smallsize image processing inHadoop Distributed File System. HDFS CombineFileInputFormat mechanism can combinemultiple files and create

InputSplits from this set of files. Inaddition to that, files whichare in the same cluster or node to be combined as InputSplit using CombineFileInputFormat mechanism. This will reduce the amountof data to be transferred from node to node andresults in general performance increases.

CombineFileInputFormat is an abstract class that does notwork with image files directly. CombinePictureInputFormat is a class that derived fromCombineFileInputFormat [23] to create CombineFileSplitas set of images. MultiImageRecordReader class is developed to create records from CombineFileSplit. This record reader uses ImageFileRecordReader class to make every imagedata as single record to map the algorithm.ImageFileOutputFormat is used to create output files from processed images and stored into HDFS.

V. ENHANCEMENT EVALUATION

HDFS cluster has been set up with 5 nodes to test the system and to evaluate the results;Sharpening jobs are performed on given set of image files on each of the cluster. HDFS cluster is setup with 5 nodes to run sharpening jobs on image sets. Each node has a Hadoop contextinstalled on a virtual machine. The performance loss in totalexecution efficiency caused due to virtualizationis mandatory but still the operations likemanagement and installation of Hadoop become easier by cloning virtualmachines. Large dynamic memory space is required by MapTasks when map-function for the imageprocessing executes.

Processing Image with large size requires more heap size in Java Virtual Machine (JVM) and hence the default size is not enough. So, maximum JVM size for Hadoop processes is increased to 700 Mb.10 different small size images are used as input image files. Scattering of the images accordingto file sizes are preserved in input folders. HDFS uses three dissimilar types of approaches to perform the sharpening job present in the input folders image files. These are (1) one task perimage approach, (2) SequenceFile processing approach and (3)Combine and Process (Parallel) images approach. The performance results are provided in figure 5.1.

The run time against file size for the two configurations; namely, the single machine sequential processing and the clustered processing is shown in figure 5.2. Four Different image processing algorithmswere used for experimentation, and different processing timings were recorded separately since each algorithm was uniquely different in its numerical processing overhead. These image processing algorithms were chosen in the Hadoop project for their recurrent use in remote sensing and for their various computational loads. More attention has been given to the processing time than the total number of images being processed. Likewise, elapsed time has been considered as the only performance degree. We compared the times taken by the single PC sequential processing and the clustered processing to observe the speedup factors and finally analyzed the results.

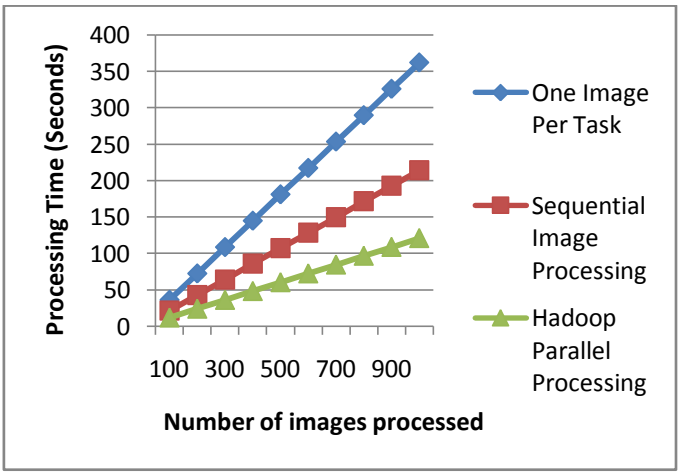

Figure 5.1 Processing Time Comparison

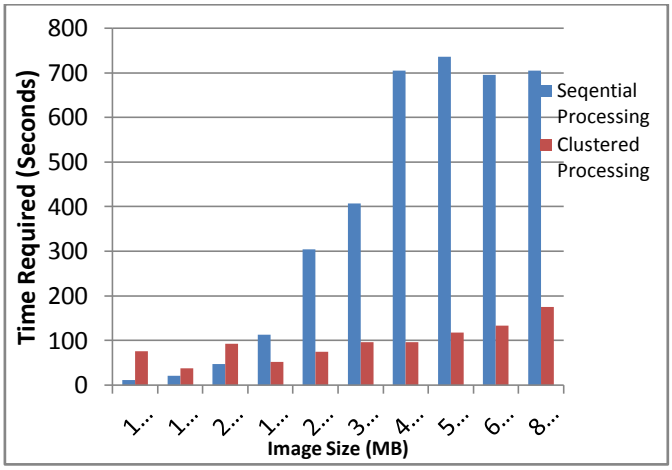

Figure 5.2 Comparison - Time Consumption vs Image Size

VI. CONCLUSION

The huge volume of visual data in recent years due to the technology development and social media networks and their need for efficient and effective processing kindle the use of distributed image processing contexts in image processing area. A case study have been presented here forimplementing parallel processing for enhancing the images in any format by using the Hadoop MapReduce structure. The experimental results have shown that the typical image processing algorithms can be efficiently parallelized with satisfactory run times when applied to remote sensing images. Huge number of images cannot be processed efficiently in the routine sequential manner. Although originally designed for text processing, HDFS MapReduce implementedon a parallel node or cluster proved suitable to process any format of images in large quantities.Results have shown that this parallel Hadoop implementation is better suited for large data sizes than the computationally intensive application. Focusing on using different image sources with different algorithms that can have a computationally demanding nature could be done in future work.

REFERENCES

[1] FACEBOOK, 2010. Facebook image storage. http://blog.facebook.com/blog.php?post=206178097130.

[2] D. Sudipto, S. Yannis, S.B. Kevin, G. Rainer, J.H. Peter, and M. John, Ricardo: Integrating R and Hadoop. *Paper presented at the 2010 international conference on Management of data*, Indianapolis, USA, June 6-11, 2010.

[3] S. Chris, L. Liu, A. Sean, and L. Jason, HIPI: A hadoop image processing interface for image-based map reduce tasks, B.S. Thesis. *University of Virginia, Department of Computer Science*, 2011.

[4] W. Keith, C. Andrew, K. Simon, G. Jeff, B. Magdalena, H. Bill, K. YongChul, and B. Yingyi, Astronomical image processing with hadoop. *Paper presented at Astronomical Data Analysis Software and Systems XX*, Boston, USA, November 7-11, 2010.

[5] http://hadoop.apache.org/.

[6] Berlinska J, M. Drozdowski (2011), Scheduling Divisible MapReduce Computations, Journal of Parallel and Distributed Computing, 71(3): 450-459.

[7] Dean J, S. Ghemawat (2010),MapReduce: A Flexible Data Processing Tool, Communications of the ACM, 53(1): 72-77.

[8] Dean J, S. Ghemawat (2008),MapReduce: Simplified Data Processing on Large Clusters,Communications of the ACM, 51(1): 1-13.

[9] Ghemawat S, H. Gobioff, S. T. Leung(2003), The Google File System, Proceedings of the 19th ACM Symposium on Operating System Principles, NY, USA: ACM, DOI:10.1145/945445.945450.

[10] Dong B et al. (2012); An Optimized Approach for Storing and Accessing Small Files on Cloud Storage, Journal of Network and Computer Applications, 35(6): 1847-1862.

[11] Dong B et al. (2010); A Novel Approach to Improving the Efficiency of Storing and Accessing Small Files on Hadoop: a Case Study by PowerPoint Files, IEEE International Conference on Services Computing (SCC), Florida, USA: IEEE, DOI:10.1109/SCC.2010.72.

[12] Krishna, M.; et al. (2010); Implementation and Performance Evaluation of a Hybrid Distributed System for Storing and Processing Images from the Web, 2nd IEEE International Conference on Cloud Computing Technology and Science, Indianapolis, USA: IEEE, 762-767, DOI:10.1109/CloudCom.2010.116.

[13] Bakshi, K. (2012) Considerations for Big Data: Architecture and Approach. Aerospace Conference-Big Sky, MT, 3-10 March 2012.

[14] Kelly, J. (2012) Big Data: Hadoop, Business Analytics and Beyond. Wikibon Whitepaper, 27 August 2012. http://wikibon.org/wiki/v/Big_Data:_Hadoop,_Business_Analytics_and_Beyond

[15] Bajcsy, P., Vandecreme, A., Amelot, J., Nguyen, P., Chalfoun, J. and Brady, M. (2013) Terabyte-Sized Image Computations on Hadoop Cluster Platforms. 2013 IEEE International Conference on Big Data, Silicon Valley, 6-9 October 2013, 729-737.

[16] Storm Project. http://storm.incubator.apache.org/

[17] Samza Project. http://samza.incubator.apache.org/

[18] http://bme.med.upatras.gr/improc/enhancement_point_processing.h

[19] Rajesh Garg, Bhawna Mittal, Sheetal Garg, "Histogram Equalization Techniques for Image Enhancement", IJECT Vol. 2, Issue 1, March2011, ISSN 2230-9543.

[20] Adaptive Recovery of Image Blocks Using Spline Approach Jong-Keuk Lee Ji-Hong Kim Jin-SeokSeo.

[21] http://wiki.apache.org/hadoop/SequenceFile

[22] Liu, X.; et al. (2009), Implementing WebGIS on Hadoop: A Case Study of Improving Small File I/O Performance on HDFS, IEEE International Conference on Cluster Computing and Workshops, Louisiana USA: IEEE, 1-8, DOI:10.1109/CLUSTR.2009.5289196.

[23] https://hadoop.apache.org/docs/current/api/org/apache/hadoop/mapred/lib/ Combine-FileInputFormat.html

