Iris Recognition on Hadoop

Thesis submitted in partial fulfillment of the

Requirements for the award of degree of

Master of Technology

In

Information Systems

Submitted By:

Shelly
(11/IS/09)

Under the Guidance of:

Prof. N.S. Raghava
Associate Professor

Department of Information Technology

Delhi Technological University

Bawana Road, Delhi – 110042

(2009-2011)

Iris Recognition on Hadoop

Thesis submitted in partial fulfillment of the

Requirements for the award of degree of

Master of Technology

In

Information Systems

Submitted By:

Shelly
(11/IS/09)

Under the Guidance of:

Prof. N.S. Raghava
Associate Professor

Department of Information Technology

Delhi Technological University

Bawana Road, Delhi – 110042

(2009-2011)

CERTIFICATE

This is to certify that **Ms. Shelly (11/IS/09)** has carried out the major project titled "**Iris Recognition on Hadoop**" as a partial requirement for the award of Master of Technology degree in Information Systems by Delhi Technological University.

The major project is a bonafide piece of work carried out and completed under my supervision and guidance during the academic session **2009-2011**. The matter contained in this report has not been submitted elsewhere for the award of any other degree.

(Project Guide)

Prof. N.S. Raghava
Associate Professor
Department of Information Technology
Delhi Technological University
Bawana Road, Delhi-110042

ACKNOWLEDGEMENT

I express my gratitude to my project guide, **Prof. N.S. Raghava, Associate Professor, Department of Information Technology, Delhi Technological University.** He provided a motivation and enthusiastic atmosphere during the many discussion we had. He has changed my thought process radically with his critical approach to problem. In spite of his hectic schedule he was always approachable and took his time off to attend to my problems and give appropriate advice. Working with him was, indeed, a great learning experience! He will always be constant source of inspiration for me.

It is an immense pleasure to express my sincere gratitude towards my **Prof. O.P.Verma, H.O.D., Department of Information Technology, Delhi Technological University,** for his immaculate guidance. I consider myself extremely fortunate to have a chance to work under his supervisions. It has been very enlightening and enjoyable experience to work under them.

I humbly extend my words of gratitude to other faculty members of this department for providing their valuable help and time whenever it was required.

Shelly
(11/IS/09)
M.Tech (Information System)

Abstract

Iris Recognition is a type of pattern recognition which recognizes a user by determining the physical structure of an individual's Iris. A unique Iris pattern is extracted from a digitized image of the eye, and encoded into an iris template. Iris template contains unique information of an individual and is stored in a database. To identify an individual by iris recognition system, an individual’s eye image is captured using video camera and converted into iris template. These templates are compared with stored iris templates in database. If templates are matches then user is said to be genuine, otherwise imposter.

Iris Recognition offers advantages over traditional recognition methods (ID cards, PIN numbers) because the person to be identified has no need to remember or carry any information. Iris pattern remains stable throughout life of a person. This characteristic makes it very attractive for use as a biometric for identifying individual.

Iris recognition is deployed for verification and/or identification in applications such as access control, border management, and Identification systems. With increasing security concerns, biometric database size is growing very fast and technologies like iris recognition has a very huge database for comparison. Iris recognition algorithms are implemented on general purpose sequential processing systems, and also existing relational database systems are not enough to handle this huge size of data in some reasonable time.

In this thesis, a parallel processing alternative using cloud computing, offering an opportunity to increase speed and use it on huge database is proposed. An open source Hadoop framework for cloud computing is used to implement the proposed system. Hadoop Distributed File System (HDFS) is used to handle large data sets, by breaking

it into blocks and replicating blocks on various machines in cloud. Template comparison is done independently on different blocks of data by various machines in parallel.

Map/Reduce programming model is used for processing large data sets. Map/Reduce process the data in <key, value> format. Iris database is stored in a <key, value> text format. Mappers process the input and produce an intermediate output. Reducer takes intermediate output and produces final result.

This research work shows how, the most time-consuming operations (matching process) of a modern iris recognition algorithm are parallelized. In particular, template matching is parallelized on a cloud based system with a demonstrated speedup gain.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Iris recognition is a method of biometric authentication based on high-resolution images of the iris of an individual's eye. A key advantage of iris recognition is iris's stability and longevity, a single enrollment can last a lifetime. [2] An iris-recognition algorithm first has to identify the approximately concentric circular outer boundaries of the iris and the pupil in a photo of an eye. The set of pixels covering only the iris is then transformed into a bit pattern that preserves the information. This bit pattern is sufficient to uniquely identify an individual. Bit pattern is also referred as Iris code or Iris template.

An iris recognition system determines a match by thresholding the distance between two aligned iris codes. It is imperative that the distances between mated iris code pairs be less than the distances between unmated pairs. A mated pair is a set of two codes from the same eye whereas an unmated pairs are a set of two codes from different eyes. Typically the mated and unmated distributions are used to visualize the potential operation of a recognition system. A good recognition system has mated scores above the decision threshold and unmated scores below the threshold [4].

The major applications of this technology so far have been: substituting for passports in automated border crossing; expediting security screening at airports; controlling access to restricted areas; and database access and login.

Before deploying Iris recognition system, an individual must be enrolled in system. As new persons keep enrolling in the system, database size is also growing very fast. To identify/verify an individual, his Iris code has to be compared against entire database which takes a lot of time and makes the system unusable in practical areas.

To handle this huge size of data set and searching in a reasonable time, a cloud computing architecture is proposed.

Cloud computing is one of the emerging technologies of today. Cloud computing promises to be limitless in the amounts of data it can analyze and to distribute data rapidly to anyone, anywhere. Cloud computing releases computer services, computer software, and data storage away from locally hosted infrastructure and into cloud-based solutions. The ability of the cloud services to provide apparently an unlimited supply of computing power on demand to users has caught the attention of industry as well as academia. In the past couple of years software providers have been moving more and more applications to the cloud [3].

Thesis Outline

In chapter 2, an overview of cloud computing architecture and Apache Hadoop Framework is described. In chapter 3, different phases of Iris recognition system is described. In chapter 4 analysis, hardware/ software requirements, and Iris Recognition flow on Hadoop is described. In Chapter 5, comparison of timing between sequential Iris Recognition and Iris recognition on Hadoop is described. Hadoop administration snap shots are also shown. In last research work is concluded and future scope is described.

Chapter 2

Cloud Computing

2.1 Definition

Cloud computing is regarded as the next trend of IT industry. Cloud computing can be loosely defined as using scalable computing resources provided as a service from outside your environment on a pay-per-use basis. You can access any of the resources that live in the "cloud" across the Internet and don't have to worry about computing capacity, bandwidth, storage, security, and reliability. Figure 2.1 [6] is a view of the layers of cloud computing and some existing offerings.

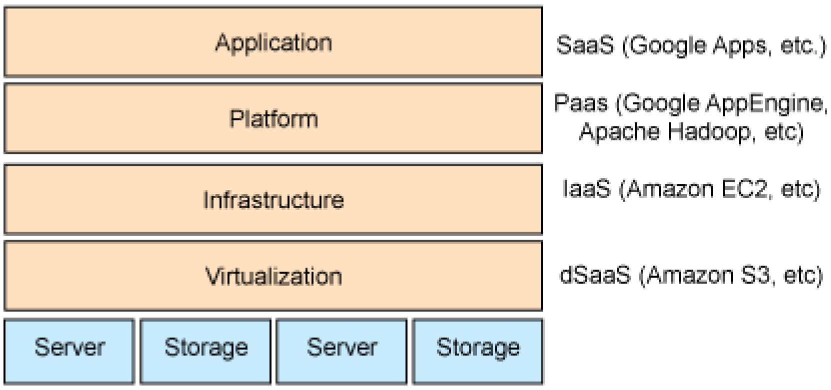


Figure 2.1: Cloud Computing Layered Architecture

The infrastructure (Infrastructure-as-a-Service,or IaaS) is the leasing of infrastructure (computing resources and storage) as a service. IaaS provides the capability for a user to lease a computer (or virtualized host) or data center with specific quality-of-service constraints that has the ability to execute certain operating systems and software. Amazon EC2 is playing a role as the IaaS in these layers and provides users virtualized hosts.

The Platform (Platform-as-a-Service or PaaS) focuses on the software framework or services, which provide the ability of APIs to "cloud" computing on the infrastructure. Apache Hadoop plays a role as PaaS and will be built on the virtualized hosts as the cloud computing platform.

The Key features of cloud computing are:

- **Service:** Everything, infrastructure, platform and software, is delivered as services.
- **Scalability:** Resources can be dynamically scalable over data center without much difficulty
- **Cost effectiveness:** It follows the “pay as use basis” model
- **Interface:** Independent of location, services are accessible as Web services via Web browsers
- **Availability**
- **Virtualization**

Cloud computing and storage convert physical resources (like processors and storage) into scalable and shareable resources over the Internet. Cloud computing gives users’ access to massive computing and storage resources without their having to know where those resources are or how they're configured. [23]

Cloud computing includes a number of technologies such as Virtualization, Web services, Service Oriented Architecture, Web 2.0, Web Mashup. One of the key new technologies used in the cloud are scalable batch processing systems. The main reference implementation here is Google Map/Reduce and its open source implementation Apache Hadoop originally developed at Yahoo. Distribution and scalability are playing important roles in the cloud, and for that Hadoop can be used. Hadoop is a cloud computing programming environment created to deal with the growing demands of modern computing and storage of massive amounts of data. Many companies, especially ones that operate through demanding websites, e.g. Amazon, Facebook, Yahoo, eHarmony and eBay, use Hadoop. [5]

2.2 Apache Hadoop

Hadoop is an open-source cloud computing environment that implements the Map/Reduce based on Google Map/Reduce. It is supported by Google, Yahoo!, and IBM, to name a few. It is a model of PaaS(Platform as a Service)[6]. Hadoop is written in Java language; any machine that supports Java can run the Hadoop software.

Hadoop is ideal on Linux as a production platform, with the framework written in the Java language. Applications on Hadoop may be developed using other languages such as C++. [22]

It has its own distributed file system called Hadoop Distributed File System (HDFS) which is based on Google File System. Hadoop uses the HDFS to divide files among several nodes, with the processor of each node only working off their own storage. Hadoop is an Apache project. Hadoop enables the development of reliable, scalable, efficient, economical and distributed computing using very simple Java interfaces - massive parallel code without the pain. HDFS based database used mainly for batch processing is HBase which is heavily inspired by Google Bigtable. The main applications for Hadoop seem to be log analysis,Web indexing, and various data mining and customer analysis applications.

At the heart of its design is the Map/Reduce implementation and HDFS (Hadoop Distributed File System), which was inspired by the Map/Reduce (introduced by a Google paper) and the Google File System.

2.3 Map/Reduce programming Model

Map/Reduce is a parallel programming model for processing large sets of data. It is a software framework introduced by Google in 2004 that supports distributed computing on large data sets on clusters of computers (or nodes). This framework is implemented in C++ with interfaces in Python and Java.. Computation takes input as a set of key/value pairs and produces output as set of key/value pairs. Goals of Map/Reduce are[25]:

– Distribution

- The data is available where needed.
- Application does not care how many computers are being used.

– Reliability

- Application does not care that computers or networks may have temporary or permanent failures.

Map/Reduce consists of two main functions Map and Reduce.

In the Map process, the master node takes the input, divides it up into smaller sub-tasks, and distributes those to worker nodes. The worker node processes that smaller task, and passes the answer back to the master node.

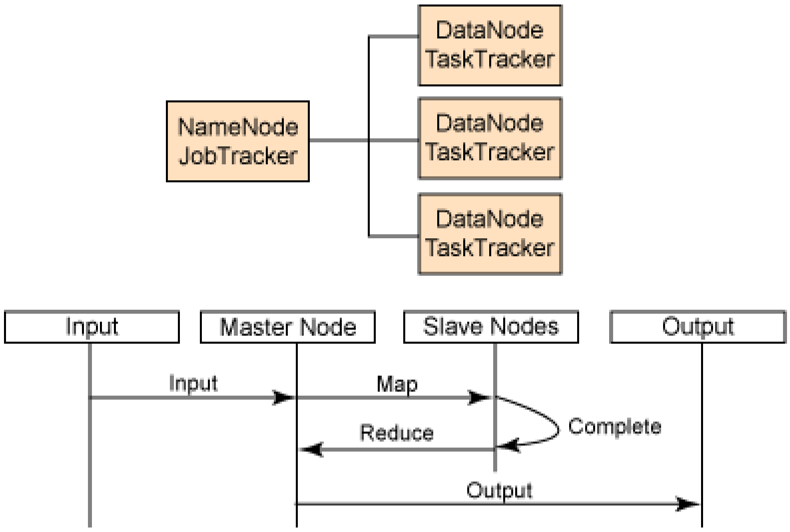


Figure 2.2 Conceptual view of Hadoop cluster and Map Reduce Flow

In the Reduce process, the master node then takes the answers of all the sub-tasks and combines them to get the output, which is the result of the original task. Figure2.2, provides a conceptual idea about the Map/Reduce flow. Computation takes input as a set of key/value pairs and produces output as set of key/value pairs.

As illustrated in Figure 2.3, Map/Reduce function call splits the input data into N pieces. Mapper maps input <key, value> pairs to a set of intermediate <key, value> pairs.

Then all the intermediate pairs are sorted by unique keys, so that all the values of the same key are grouped together. The reducer reduces this grouped set of intermediate values. The outputs of all reduce functions are merged to produce the output of the complete Map/Reduce job.

The advantage of Map/Reduce is that it allows for the distributed processing of the map and reduction operations. Because each mapping operation is independent, all maps can be performed in parallel, thus reducing the total computing time.

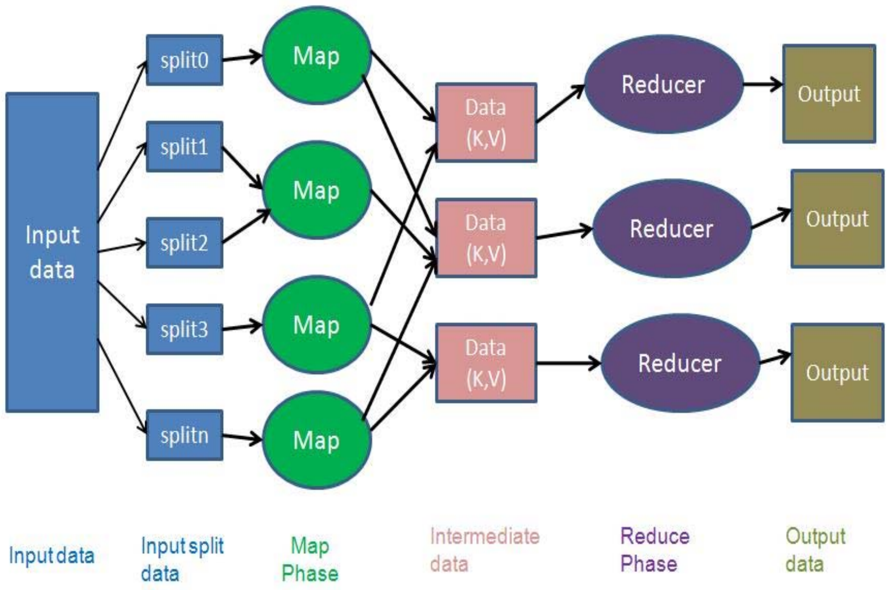


Figure 2.3 Map Reduce Framework

2.4 Hadoop Distributed File System

Hadoop comes with a file system named HDFS. HDFS stands for Hadoop Distribution File System. But it is significantly different from distributed systems. It is highly fault tolerant and designed to deploy on low cost hardware. HDFS is designed to be highly portable. HDFS provides high throughput access to application data. [21] It is designed to hold large amount of data. Hadoop takes any input from HDFS and stores results on HDFS. HDFS is part of the Apache Hadoop project, which is part of the Apache Lucene project. [21]

The Hadoop file system is designed for storing petabytes of a file with streaming data access using the idea that most efficient data processing pattern is a write-once, read-many-times pattern. HDFS stores Metadata on a dedicated server, called NameNode. Application data are stored on other servers called DataNodes. All the servers are fully connected and communicate with each other using TCP based protocols.

2.4.1 Hadoop Distributed File System Features

- **Data Disk Failure, Heartbeats and Re-Replication**

The primary objective of the HDFS is to store data reliably even in the presence of failures. The three common types of failures are NameNode failures, DataNodes failures and network partitions. [8]

NameNode considers DataNodes as alive as long as it receives Heartbeat message (default Heartbeat interval is three seconds) from DataNodes. If the NameNode does not receive a heartbeat from a DataNodes in ten minutes the NameNode considers the DataNodes as dead and stop forwarding IO request to it. The NameNode then schedules the creation of new replicas of those blocks on other DataNodes. Heartbeats carry information about total storage capacity, fraction of storage in use, and the number of data transfers currently in progress. These statistics are used for the NameNode's space allocation and load balancing decisions. The NameNode can process thousands of heartbeats per second without affecting other NameNode operations.

- **Cluster Rebalancing**

The HDFS architecture has data rebalancing schemes in which data is automatically moved from one DataNode to another if the free space threshold is reached. In the event of a sudden high demand for a particular file, a scheme might dynamically create additional replicas and rebalance other data in the cluster. These types of data rebalancing schemes are not yet implemented. [8]

- **Data Integrity**

Block of data can be corrupted due to many reasons such as network faults, buggy software or faults in a storage device. So, at the time of file creation checksum is used and stored for each block. While retrieving a file, it is first verified with those checksums and if verification fails, then another replica of data is used.

- **Metadata Disk Failure**

Corrupted Fsimage and the EditLog may stop the HDFS functioning. For redundancy, NameNode is configured to have multiple copies of these files and are updated synchronously.

- **Snapshots**

A Snapshot saves the current state of the file system at any instance of time. The main usage of this feature is to rollback to the previous state if the upgrading resulted in data loss or corruption.

- **Communication Protocols**

All HDFS communication protocols are layered on top of the TCP/IP protocol. A client establishes a connection to a configurable TCP port on the NameNode machine. It talks the Client Protocol with the NameNode. The DataNodes talk to the

NameNode using the DataNodes Protocol. A Remote Procedure Call (RPC) abstraction wraps both the Client Protocol and the DataNodes Protocol. By design, the NameNode never initiates any RPCs. Instead, it only responds to RPC requests issued by DataNodes or clients. [8]

- **Highly fault-tolerant**
- **High throughput**
- **Suitable for applications with large data sets**
- **Streaming access to file system data**
- **Can be built out of commodity hardware**

2.4.2 Architecture

HDFS has master/slave architecture. A HDFS cluster consists of, Figure 2.4 [8] shows, a NameNode (master node) and a number of DataNodes (slave nodes).The NameNode and DataNodes are pieces of software designed to run on commodity machines. The usage of the highly portable Java language means that HDFS can be deployed on a wide range of machines. A typical deployment has a dedicated machine that runs only the NameNode software. Each of the other machines in the cluster runs one instance of the DataNodes software. The architecture does not preclude running multiple DataNodes on the same machine but in a real deployment one machine usually runs one DataNode.

NameNode

NameNode manages the file system namespace, Metadata for all the files and directories in the tree. It regulates access of files to its entire client. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes. As there is low amount of Metadata per file (only file names, permissions, and the locations of each block of each file), all of this information can be stored in the main memory of the NameNode machine, which allow fast access to the Metadata. To open a file, client reads Metadata from NameNode and gets the list of blocks, which comprise a file. It gets the location of DataNodes which stores these blocks and client can access it, possibly, in parallel.

As the NameNode machine can also get fail so this Metadata must be preserved. A log of this Metadata file is stored on another machine, called, Secondary NameNode. [26]

DataNode

In addition, there are a number of DataNodes, usually one per node in the cluster, which manage storage attached to the nodes that they run on. Each machine/node on cluster is referred to as DataNode. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes

DataNodes store and retrieve blocks when requested (by clients or the NameNode), and they report back to the NameNode periodically with lists of blocks they are storing. The DataNodes are responsible for serving read and write requests from the file system's clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode [8]. These blocks are stored in replication on different nodes in cluster, so that if one node fails other nodes can take load.

DataNodes and NameNode connections are established by handshake where namespace ID and the software version of the DataNodes are verified. The namespace ID is assigned to the file system instance when it is formatted. The namespace ID is stored persistently on all nodes of the cluster. A different namespace ID node cannot join the cluster [27]. A new DataNode without any namespace ID can join the cluster and receive the cluster's namespace ID and DataNode registers with the NameNode with storage ID. A DataNode identifies block replicas in its possession to the NameNode by sending a block report. A block report contains the block id, the generation stamp and the length for each block replica the server hosts. The first block report is sent immediately after the DataNodes registrations. Subsequent block reports are sent every hour and provide the NameNode with an up-to date view of where block replicas are located on the cluster. [7]

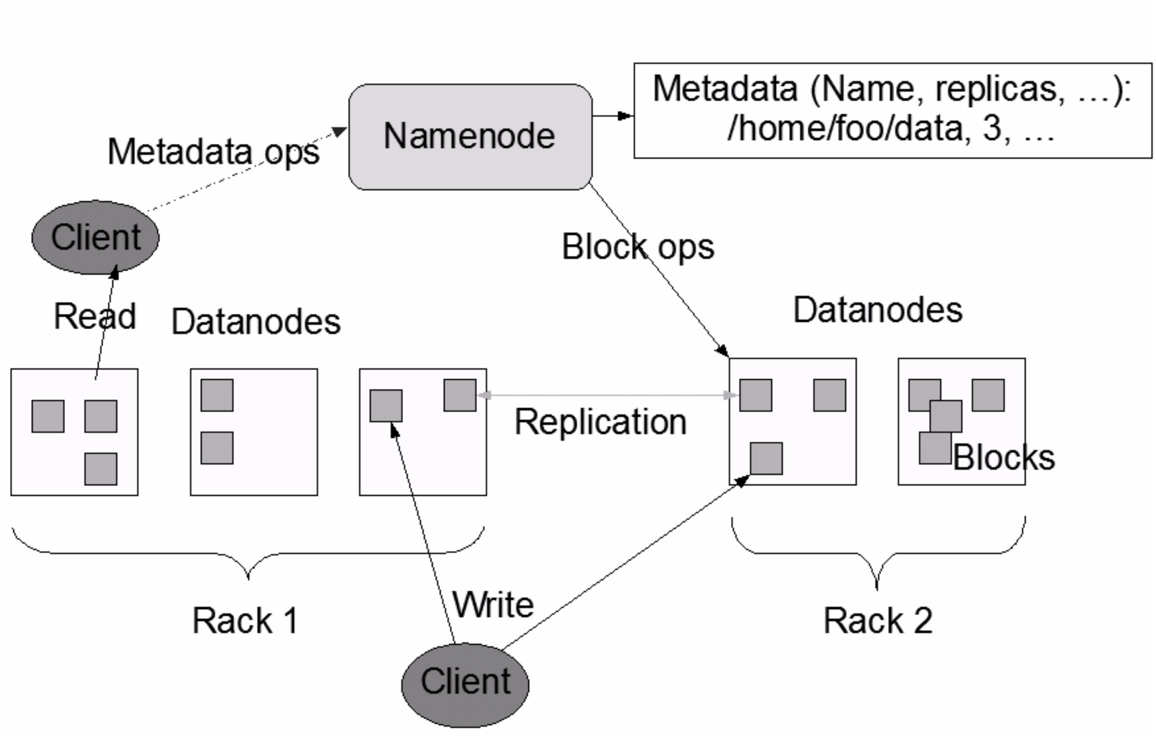


Figure 2.4: HDFS Architecture

JobTracker and TaskTracker

When an application is submitted, input and output directories contained in the HDFS should be provided. The JobTracker, as the single control point for launching the Map/Reduce applications, decides how many TaskTracker and subordinate tasks to be created and then assigns each sub-task to TaskTracker. Each TaskTracker reports status and completed tasks back to the JobTracker. As tasks run, the TaskTracker reports status
to the JobTracker every 10 seconds.[25]

Usually one master node acts as the NameNode and JobTracker and the slave acts as the DataNode and TaskTracker. The conceptual view of Hadoop cluster and the follow of Map/Reduce are shown in Figure 2.

2.4.3 Hadoop Distributed File System client

Reading a file

To read a file, HDFS client first contacts NameNode. It returns list of addresses of the DataNodes that have a copy of the blocks of the file. Then client connects to the closest DataNodes directly for each block and requests the transfer of the desired block. Figure 2.5 [7] shows the main sequence of events involved in reading data from HDFS.

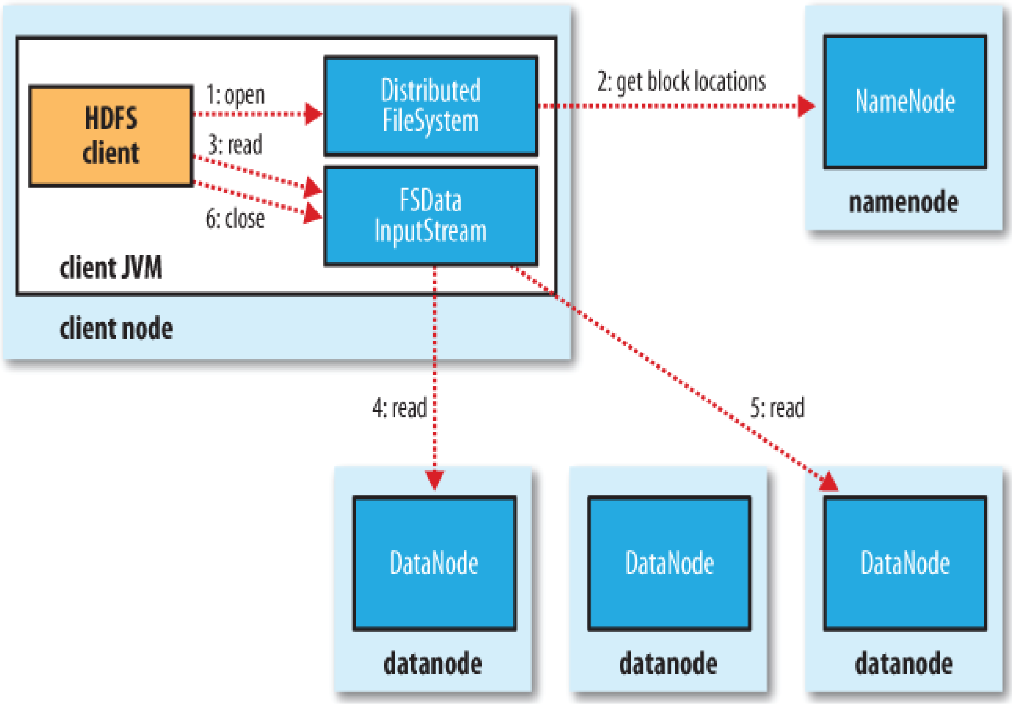
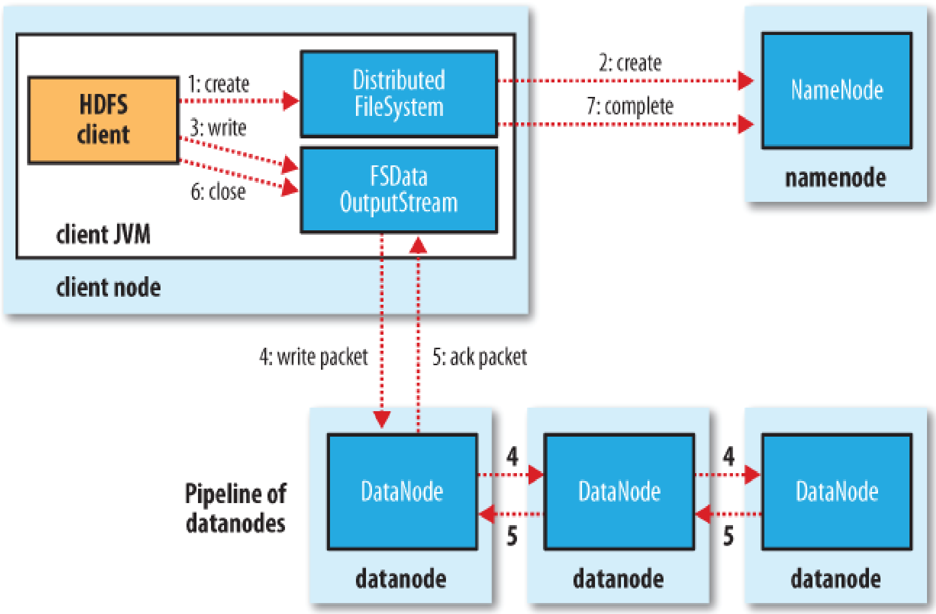


Figure 2.5: A Client Reading Data from HDFS

Writing to a File

For writing to a file, HDFS client first creates an empty file without any blocks. File creation is only possible when the client has writing permission and a new file does not exist in the system. NameNode records new file creation and allocates data blocks to list of suitable DataNodes to host replicas of the first block of the file. Replication of data makes DataNodes in pipeline. When the first block is filled, new DataNodes are requested to host replicas of the next block. A new pipeline is organized, and the client sends the further bytes of the file. Each choice of DataNodes is likely to be different. If a DataNode in pipeline fails while writing the data then pipeline is first closed and partial block on failed data node is deleted and failed DataNode is removed from the pipeline. New DataNodes in the pipeline are chosen to write remaining blocks of data. Figure 2.6[7] shows the steps involved in creating a new file, writing data to it and then closing the file.



4

Figure2.6: A Client Writing Data to HDFS

2.4.4 Replication Management

The NameNode is responsible for block replication. Replica placement determines HDFS reliability, availability and performance. Each replica on unique racks helps in preventing data loses on entire rack failure and allows use of bandwidth from multiple racks when reading data. This policy evenly distributes replicas in the cluster which makes it easy to balance load on component failure. However, this policy increases the cost of writes because a write needs to transfer blocks to multiple racks. The NameNode keeps checking the number of replicas. If a block is under replication, then it is put in the replication priority queue. The highest priority is given to low replica value. Placement of new replica is also based on priority of replication. If the number of existing replicas is one, then a different rack is chosen to place the next replica. In case of two replicas of the block on the same rack, the third replica is placed on a different rack. Otherwise, the third replica is placed on a different node in the same rack as an existing replica.

The NameNode also checks that all replica of a block should not be at one rack. If so, NameNode treats the block as under-replicated and replicates the block to a different rack and deletes the old replica.

2.4.5 Advantages and Disadvantages

Advantages of the HDFS are:

- **Reliable Storage** - HDFS is a fault tolerant storage system. HDFS can significantly store huge amounts of data, scale up incrementally and can effectively handle the failure of significant parts of the storage infrastructure without losing data.
- **Commodity Hardware -** HDFS is designed to run on highly unreliable hardware and so is less expensive compared to other fault tolerant storage systems.
- **Distributed -** HDFS data are distributed over many nodes in a cluster and so parallel analysis are possible and this eliminates the bottlenecks imposed by monolithic storage systems.
- **Availability -** Block replication is one of the main features of HDFS. By default each block is replicated by the client to three DataNodes but replication factor can be configured more then 3 at creation time. Because of replication HDFS provides high availability of data in high demand.

Limitations in HDFS are:

- **Architectural Bottlenecks -** There are scheduling delays in the Hadoop architecture that result in cluster nodes waiting for new tasks. More over disk is not used in a streaming manner, the access pattern is periodic. HDFS client serializes computation and I/O instead of decoupling and pipelining those operations.
- **Portability Limitations -** HDFS being in Java could not able to support some performance-enhancing features in the native file system.
- **Small File -** HDFS is not efficient for large numbers of small files.
- **Single Master Node –** There might be risk of data loss because of single NameNode.
- **Latency Data Access –** At the expense of latency HDFS delivers a high throughput of data. If an application needs low–latency access to data then HDFS is not a good choice.

One of the major advantages of the HDFS is scalability. Besides its limitations, HDFS is highly in demand when data sets are very large where scalability plays an important role.

Chapter 3

Iris Recognition

3.1 Introduction

A biometric system provides automatic recognition of an individual based on some sort of unique feature or characteristic possessed by the individual.[30] Biometric systems have been developed based on fingerprints, facial features, voice, hand geometry, handwriting, the retina [12], and the iris.

Biometric systems work by first capturing a sample of the feature, such as recording a digital sound signal for voice recognition, or taking a digital color image for face recognition. The sample is then transformed using some sort of mathematical function into a biometric template. The biometric template will provide a normalized, efficient and highly discriminating representation of the feature, which can then be objectively compared with other templates in order to determine identity. Most biometric systems allow two modes of operation. An enrollment mode for adding templates to a database, and an identification mode, where a template is created for an individual and then a match is searched for in the database of pre-enrolled templates.

A good biometric is characterized by use of a feature that is; highly unique – so that the chance of any two people having the same characteristic will be minimal, stable so that the feature does not change over time, and be easily captured in order to provide convenience to the user, and prevent misrepresentation of the feature.

3.2 The Human Iris

The iris is a thin circular diaphragm, which lies between the cornea and the lens of the human eye. A front-on view of the iris is shown in Figure 3.1. The iris is perforated close to its center by a circular aperture known as the pupil. The function of the iris is to control the amount of light entering through the pupil, and this is done by the sphincter and the dilator muscles, which adjust the size of the pupil. The average diameter of the iris is 12 mm, and the pupil size can vary from 10% to 80% of the iris diameter [13].

The iris consists of a number of layers; the lowest is the epithelium layer, which contains dense pigmentation cells. The stromal layer lies above the epithelium layer, and contains blood vessels, pigment cells and the two iris muscles. The density of stromal pigmentation determines the color of the iris. The externally visible surface of the multilayered iris contains two zones, which often differ in color [14]. An outer ciliary zone and an inner pupillary zone, and these two zones are divided by the collarette – which appears as a zigzag pattern.

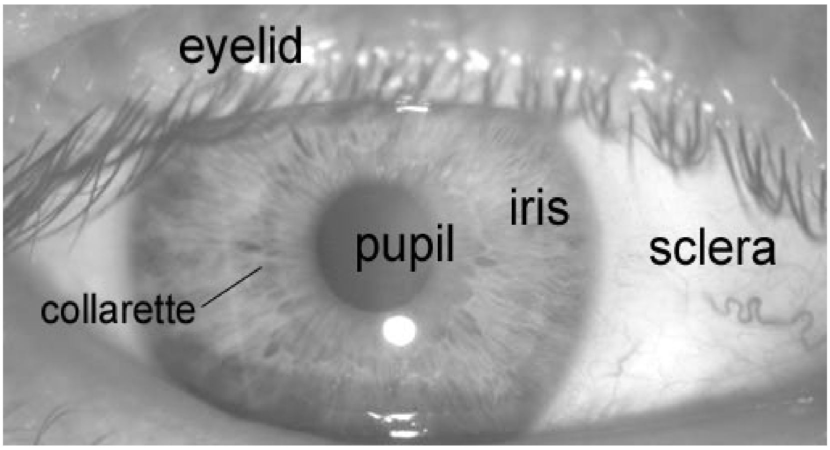


Figure3.1: A Front on View of Human Eye

Formation of the iris begins during the third month of embryonic life [14]. The unique pattern on the surface of the iris is formed during the first year of life, and pigmentation of the stroma takes place for the first few years. Formation of the unique patterns of the iris is random and not related to any genetic factors [15]. The only characteristic that is dependent on genetics is the pigmentation of the iris, which determines its color. Due to the epigenetic nature of iris patterns, the two eyes of an individual contain completely independent iris patterns, and identical twins possess uncorrelated iris patterns. For further details on the anatomy of the human eye consult the book by Wolff [14].

3.3 Iris Recognition

The iris is an externally visible, yet protected organ whose unique pattern remains stable throughout adult life. These characteristics make it very attractive for use as a biometric for identifying individuals. Image processing techniques can be employed to extract the unique iris pattern from a digitized image of the eye, and encode it into a biometric template, which can be stored in a database. This biometric template contains an objective mathematical representation of the unique information stored in the iris, and allows comparisons to be made between templates. When a subject wishes to be identified by iris recognition system, their eye is first photographed, and then a template created for their iris region. This template is then compared with the other templates stored in a database until either a matching template is found and the subject is identified, or no match is found and the subject remains unidentified.

The system is to be composed of a number of sub-systems, which correspond to each stage of iris recognition.

These stages are Image acquisition- deals with capturing sequence of iris images from the subject using cameras and sensors, Preprocessing -involves various steps such as iris liveness detection, pupil and iris boundary detection, eyelid detection and removal and normalization, Feature encoding – creating a template containing only the most discriminating features of the iris Feature Matching (Recognition) - achieves result by comparison of features with stored patterns. The input to the system will be an eye image, and the output will be an iris template, which will provide a mathematical representation of the iris region.

3.3.1 Image Acquisition

Image acquisition deals with capturing sequence of iris images from the subject using cameras and sensors. An image acquisition consists of illumination, position and physical capture system. The occlusion, lighting, number of pixels on the iris are factors that affect the image quality [15]. Many iris recognition systems require stern cooperation of the user for image acquisition. Ketchantang [16] proposed a method in which the entire sequence of images is acquired during enrollment and the best feasible images are selected, to increase flexibility. Enrollment aids to provide strong identity management.

3.3.2 Preprocessing

The purpose of preprocessing stage of iris recognition is to isolate the actual iris region in a digital eye image. The iris region, shown in Figure 1.1, can be approximated by two circles, one for the iris/sclera boundary and another, interior to the first, for the iris/pupil boundary. The eyelids and eyelashes normally occlude the upper and lower parts of the iris region. Also, specular reflections can occur within the iris region corrupting the iris pattern. A technique is required to isolate and exclude these arte facts as well as locating the circular iris region.

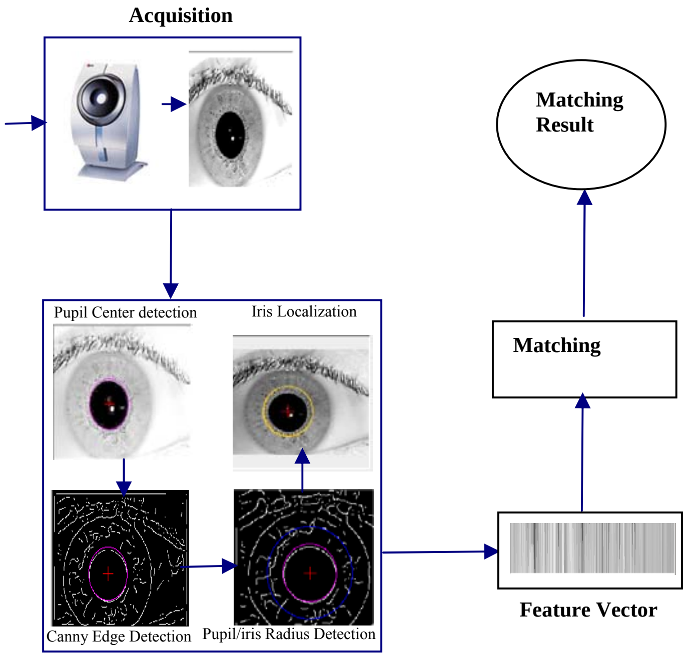


Figure 3.2 Iris Recognition Process

Hough Transform

The Hough transform is a standard computer vision algorithm that can be used to determine the parameters of simple geometric objects, such as lines and circles, present in an image. The circular Hough transform can be employed to deduce the radius and centre coordinates of the pupil and iris regions. An automatic segmentation algorithm based on the circular Hough transform is employed by Wildes et al. [17], Kong and Zhang [19], Tisse et al. [18], and Ma et al. [20]. Firstly, an edge map is generated by calculating the first derivatives of intensity values in an eye image and then thresholding the result. From the edge map, votes are cast in Hough space for the parameters of circles passing through each edge point. These parameters are the centre coordinates xc and yc, and the radius r, which are able to define any circle according to the equation

$$xc^2 + y\,c^2 - r^2 = 0 \qquad (3.1)$$

A maximum point in the Hough space will correspond to the radius and center coordinates of the circle best defined by the edge points. Wildes et al. and Kong and Zhang also make use of the parabolic Hough transform to detect the eyelids, approximating the upper and lower eyelids with parabolic arcs, which are represented as;

$$(-(x - h_j)\sin\theta_j + (y - k_j)\cos\theta_j)2 = a_j((x - h_j)\cos\theta_j + (y - k_j)\sin\theta_j) \quad (3.2)$$

Where a_j controls the curvature, (h_j, k_j) is the peak of the parabola and θ_j is the angle of rotation relative to the x-axis.

In performing the preceding edge detection step, Wildes et al. bias the derivatives in the horizontal direction for detecting the eyelids, and in the vertical direction for detecting the outer circular boundary of the iris, this is illustrated in Figure 2.1. The motivation for this is that the eyelids are usually horizontally aligned, and also the eyelid edge map will corrupt the circular iris boundary edge map if using all gradient data. Taking only the vertical gradients for locating the iris boundary will reduce influence of the eyelids when performing circular Hough transform, and not all of the edge pixels defining the circle are required for successful localization. Not only does this make circle localization more accurate, it also makes it more efficient, since there are less edge points to cast votes in the Hough space.

There are a number of problems with the Hough transform method. First of all, it requires threshold values to be chosen for edge detection, and this may result in critical edge points being removed, resulting in failure to detect circles/arcs. Secondly, the Hough transform is computationally intensive due to its 'brute-force' approach, and thus may not be suitable for real time applications.

It was decided to use circular Hough transform for detecting the iris and pupil boundaries. This involves first employing canny edge detection to generate an edge map. Gradients were biased in the vertical direction for the outer iris/sclera boundary, as suggested by Wildes et al. [19]. Vertical and horizontal gradients were weighted equally for the inner iris/pupil boundary.

3.3.3 Feature Extraction

In order to provide accurate recognition of individuals, the most discriminating information present in an iris pattern must be extracted. Only the significant features of the iris must be encoded so that comparisons between templates can be made [36]. Most iris recognition systems make use of a band pass decomposition of the iris image to create a biometric template.

Feature extraction identifies the most prominent features for classification. Some of the features are x-y coordinates, radius, shape and size of the pupil, intensity values, orientation of the pupil ellipse and ratio between average intensity of two pupils. The features are encoded to a format suitable for recognition.

3.3.4 Feature Matching

The template that is generated in the feature encoding process will also need a corresponding matching metric, which gives a measure of similarity between two iris templates. This metric should give one range of values when comparing templates generated from the same eye, known as intra-class comparisons, and another range of values when comparing templates created from different irises, known as inter-class comparisons. These two cases should give distinct and separate values, so that a decision can be made with high confidence as to whether two templates are from the same iris, or from two different irises.

3.4 Strengths and weaknesses [10]

Strengths	**Weaknesses**
It has the potential for exceptionally high levels of accuracy	Acquisition of the image requires moderate training and attentiveness
It is capable of reliable identification as well as verification	It has a propensity for false rejection
It maintains stability of characteristic over a lifetime.	A proprietary acquisition device is necessary for deployment
	There is some user discomfort with eye-based technology

3.5 Advantages and Disadvantages [10]

Advantages

• Iris is currently claimed and perhaps widely believed to be the most accurate biometric, especially when it comes to FA rates. Iris has very few False Accepts (the important security aspect). Therefore iris might be a good biometric for pure identification applications

• Given that the iris sample acquisition process is solved using unobtrusive and distant cameras, the sensing of the biometric is without physical contact and without too much inconvenience

• Iris has received little negative press and may therefore be more readily accepted. The fact that there is no criminal association helps

• The dominant commercial vendors claim that iris does not involve high training costs.

Disadvantages

• Since the iris is small, sampling the iris pattern requires much user cooperation or complex, expensive input devices

• The performance of iris authentication may be impaired by glasses, sunglasses, and contact lenses; subjects may have to remove them

• The iris biometric, in general, is not left as evidence on the scene of crime; no trace left

• Some people are missing one or both eyes, while others do not have the motor control necessary to reliably enroll in such a system

The physiological properties of irises are major advantages to using them as a method of authentication. As discussed earlier, the morphogenesis of the iris that occurs during the seventh month of gestation results in the uniqueness of the iris even between multi-birth children. These patterns remain stable throughout life and are protected by the body's own mechanisms. This randomness in irises makes them very difficult to forge and hence imitate the actual person.

In addition to the physiological benefits, iris-scanning technology is not very intrusive as there is no direct contact between the subject and the camera technology. It is noninvasive, as it does not use any laser technology, just simple video technology. The camera does not record an image unless the user actually engages it. It poses no difficulty in enrolling people that wear glasses or contact lenses.

The accurateness of the scanning technology is a major benefit with error rates being very low, hence resulting in a highly reliable system for authentication. Scalability and speed of the technology are a major advantage. The technology is designed to be used with large-scale applications such as with ATM's. The speed of the database iris records are stored in is very important. Users do not like spending a lot of time being authenticated and the ability of the system to scan and compare the iris within a matter of minutes is a major benefit.

3.6 Iris Practical Applications [11]

Iris recognition systems are being used today to control physical access, to facilitate identity verification and for computer authentication. Real-world iris recognition applications have been implemented for airport and prison security, automatic teller machines, authentication using single sign-on, to replace ID cards, and to secure schools and hospitals.

Based on the applications, reliability, ease of use, and software and hardware devices that currently support it, iris recognition technology has potential for widespread use. Iris recognition costs compare favorably with many other biometric products on the market today. Next to retinal scanning, iris recognition is the most secure biometric technology available. Iris recognition removes the need for physical contact with the biometric recording device and is recommended for both verification and identification.

The increase in requirements for securing airports could drive up the use of biometric devices for transportation security. Iris recognition systems seem to fill that need well and there is already evidence that the transportation industry recognizes the usefulness of iris recognition.

Terrorist activity increases the need for secure access to restricted areas, so there may also be increases in installation of biometric devices for building entry. In addition to its reliability, the lack of physical contact required for verification may make iris recognition more attractive to the general public than fingerprint or hand geometry biometric devices.

Chapter 4

Proposed System

4.1 Analysis

Most biometric systems (such as Iris) must

- Collect the primary data (images or samples)
- Analyze the data (extract features)
- Use the data (searching/matching)
- Store the data (images and indexes)
- Exchange the data (sharing)

Each of these steps requires varying amounts of computation and storage resources.

Building and testing large scale biometric algorithms involved a large group of computers, ad-hoc testing scripts, and a group of system administrators trying to maintain the whole assembly.

The Basic Problem: ***too much data, too much time.*** [27]

- Training biometric algorithms requires large training sets.
- Score results are quadratic in growth rate: N cases gives N*N results.
- Multiple matching algorithms can be used as input, so again, more data.
- Multiple feature extraction algorithms each produce their own feature data from input images.
- The more permutations are involved, the more CPU time and storage is needed.

In biometric testing, there is a lot of data to manage and handle reliably. Tests need to be fully checkpointed, reliable, and redundant. A single hard disk or computer failure could doom an entire test, so the framework that runs the tests needs to be robust and automatically handle the problems

With the advent in on-demand cloud based computing services and frameworks; we can now scale resources as needed, in a reliable, redundant and robust fashion. Thousands of CPU's can be allocated as needed for the test in order to meet throughput or latency requirements. Multi-Terabyte databases are no problem [29]. Processing the statistics for trillions of match scores is easily done.

Rather than using a conventional group of networked machines, we have borrowed the technology used by Google, Yahoo, and Amazon for search processing. Data and processing are distributed throughout a network of provisioned on-demand computers.

An advanced scheduling system based on Apache Hadoop breaks up the calculations into small jobs that are scheduled throughout a loosely coupled network. A highly reliable filesystem replicates data in at least three places in the network, providing reliability in the face of a disk or network failure. Data replication not only helps with reliability, but increases data read throughput; important for feature matching.

A record oriented database is used to store data. The data within each record is distributed among many different network nodes. Processing nodes automatically get copies of the image data as the biometric records are added to the system. The network wide distributed storage in the file system means that biometric record databases can scale from 1 to 100 million records with ease.

4.2 Development Environment

Seven Linux servers were used to build a cluster for cloud environment. Hadoop framework were installed at each server in which one server node is acting as master (as NameNode) and others are slaves (as DataNodes).

Hadoop engine was deployed over the seven node cluster and Java runtime was setup to use the common Hadoop configuration, as specified by the NameNode (master node) in the cluster

Software and Language Versions

Table 4.1 software Description

Software/Language	Version
Operating System	Red Hat Linux (64 bit) with Kernel 2.6.9-42.ELsmp
Hadoop Version	0.21.0
Java Platform	1.6.0

Hardware Specification of Each Hadoop Node

Table 4.2 Hardware Description

Hardware Components	Description
Processor	Dual core AMD 2200 series
CPU speed	2.8GHZ
No. of cores/ processor	4
Main Memory	16GB
Hard Disk Space	1 TB

4.3 System Architecture

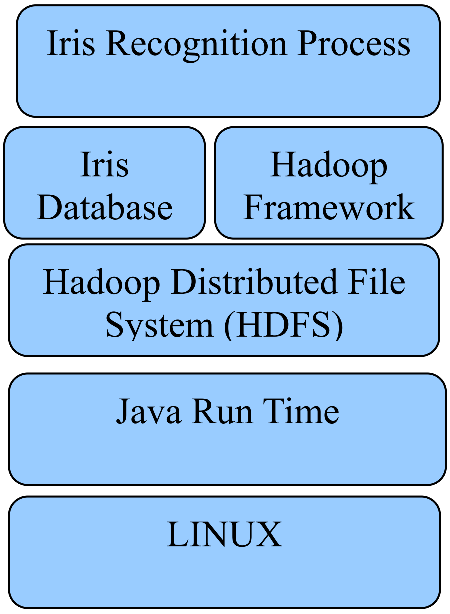


Figure 4.1 System Architecture

As shown in Figure 4.1, Linux is used as an operating system and Java runtime Environment is installed on top of it. Other main components of the System are explained as follows:

- Hadoop is installed on Java Runtime Environment.
- HDFS stores input and output files.
- Map Reduce framework has been used to calculate the matching probability values of database template against iris template of input image.
- Iris recognition process is developed using java and Eclipse as IDE.
- Eclipse is configured to support the Hadoop framework.

Map/Reduce Tools for Eclipse is a plug-in that supports the creation of Map/Reduce applications within Eclipse. It uses the Hadoop open-source Map/Reduce framework, which enables data-intensive applications to run on large clusters of commodity hardware. The plug-in supports the following features from within Eclipse: [24]

- The ability to package and deploy a Java project as a JAR (Java Archive) file to a Hadoop server (local and remote).
- A separate perspective with a view of Hadoop servers, the Hadoop distributed file system (HDFS), and current job status.
- Wizards for facilitating the development of classes based on the Map/Reduce framework.
- This plug-in runs on Windows, Linux, or any system that can run Eclipse.

4.4 Iris Recognition on Hadoop

In this paper we have run iris matching process on Map/Reduce framework. Map/Reduce framework works on <key, value> format. As Hadoop works on the record format, so entire database is stored in a text format and uploaded on HDFS. Mappers take its input from HDFS and reducer stores the final result on HDFS.

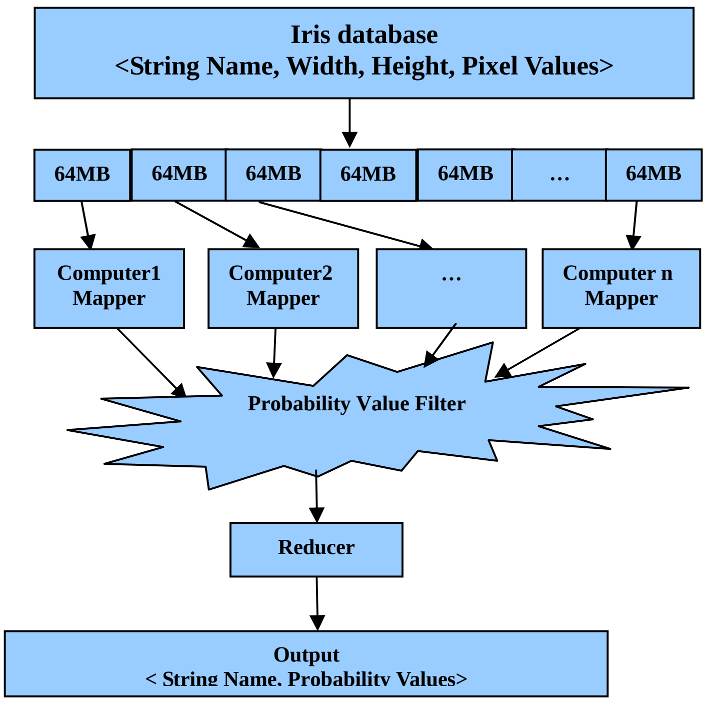


Figure 4.2 Iris Recognition Flow on Hadoop Framework

The format of record is as follows:

<String name, width, height, pixel values>.

String name is used as key. Pixel values are the intensity values of the image. Width and height parameter determines how many values to be read against one key.

IITD [31], MMU [32], UBIRIS [33] and CASIA [34] Iris images are used to create this database.

The input image that has to be matched is scanned, template is extracted and then probability is calculated of each stored template against input image. The entire database is divided into chunks and distributed to all mappers. The output of each mapper is**<string name, probability value>**. These intermediate outputs are collected from all mappers, sorting and shuffling is applied. A threshold value is selected. If probability value is greater than threshold value then given to reducer task as input.

Reducer sorts the probability values in increasing order and stores the result in an output file.

Chapter 5

Results Analysis

For the result analysis we have setup two versions of Iris Recognition application and use it on the same data set. The first version is a sequential execution, while other version is the Hadoop version of the Iris Recognition application. We tested and compare both versions with different database size. The timing (in seconds) of all runs is shown in Table 5.1.

Table 5.1 Timing results of iris matching on different database size.

File size	Sequential Execution Time (in seconds)	Hadoop Execution Time (in seconds)	Speedup	Efficiency
1.1GB	60	44	1.36	0.17
2.1 GB	202	49	4.12	0.515
4.2GB	422	77	5.48	0.685
8.5GB	850	128	6.64	0.83
10.9GB	1062	165	6.436	0.804
13.2GB	1276	192	6.645	0.83

Here, we use the widely used formulation in calculating speedup and efficiency of systems as shown [35]

$$SpeedUp = \frac{SequentialExecutionTime}{ParallelExecutionTime}$$

$$Efficiency = \frac{SpeedUp}{No.of\ ProcessingNodes}$$

From table 5.1 results, we can see as the size of data set grows speedup and efficiency of Hadoop framework increases. The Hadoop performance is non-linear, but sequential execution timing is almost linear, as shown in Figure 5.1.

We observe from Table 5.1 that speedup gain after 8.5GB is not as much as the previous ones. This is due to the insufficient number of compute nodes as number of map tasks increases with growing dataset size.

Figure 5.1: Bar Chart Comparing Execution Time of Iris Recognition System

Figure 5.2 shows the number of map tasks created at every run with different data set size.

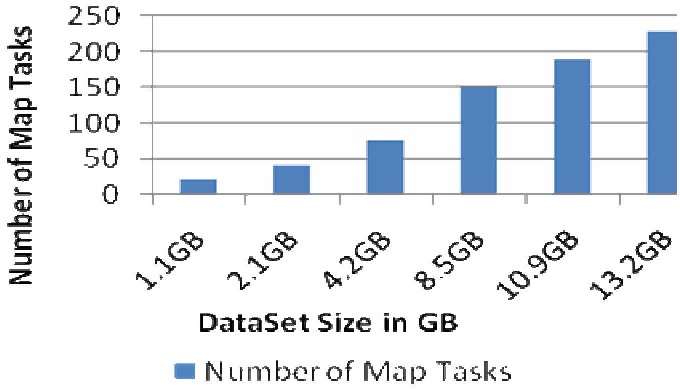


Figure 5.2: Bar Chart for Number of Map Tasks with various sizes of Dataset.

Hadoop Administration Snap Shots

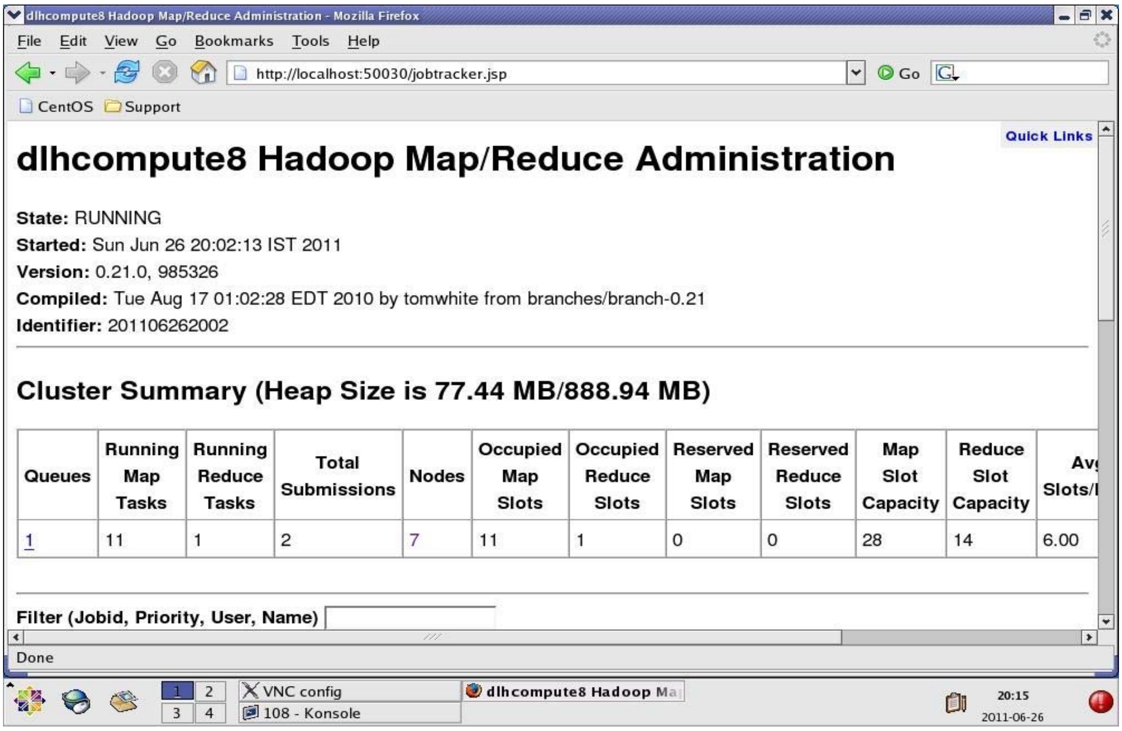


Figure 5.3: (a) Running Number of Map/Reduce Tasks, No. of Nodes etc.

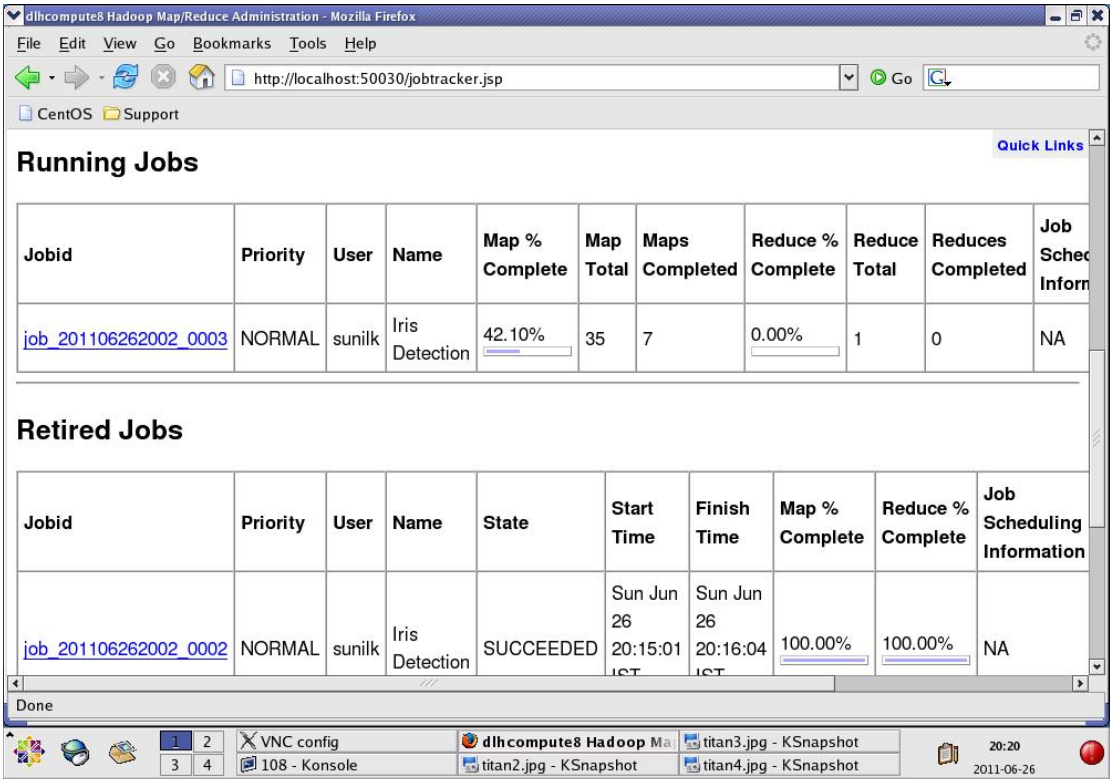
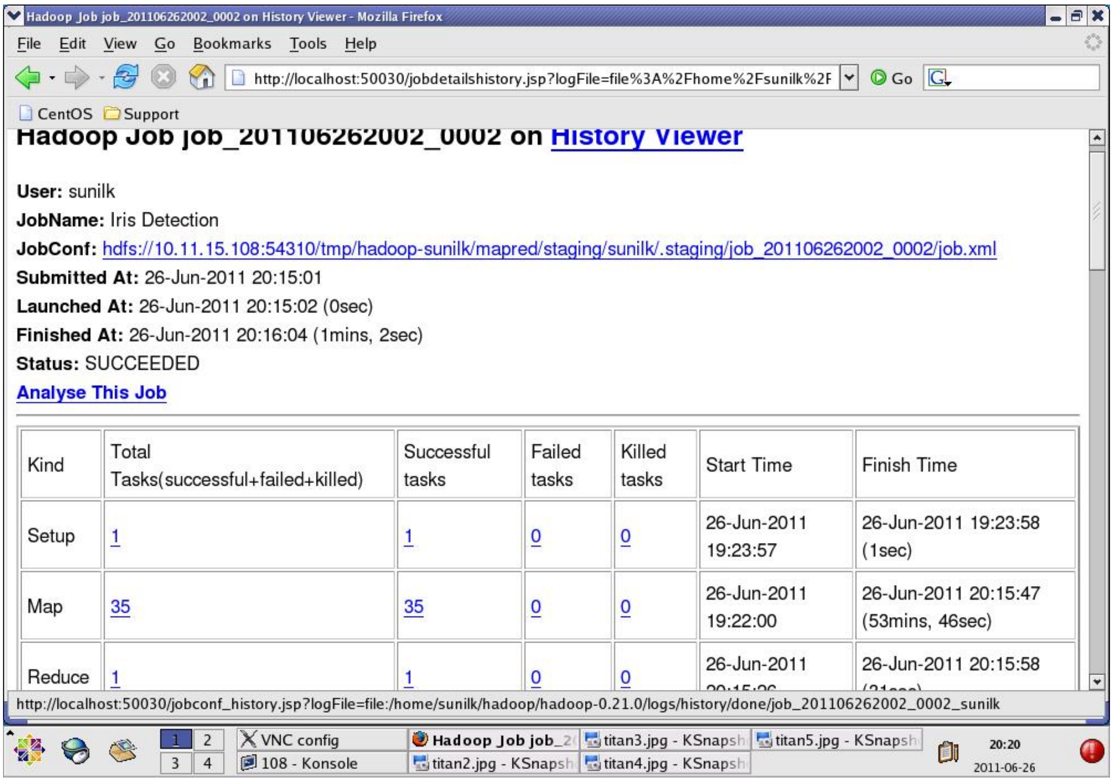
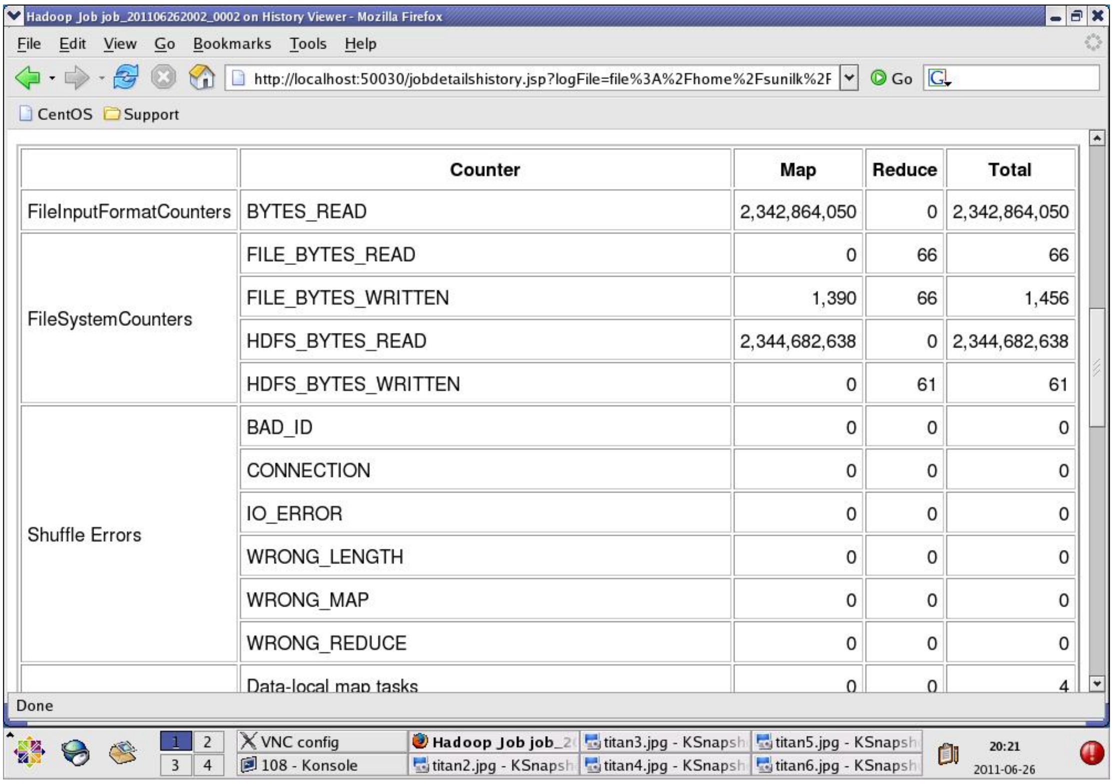


Figure 5.3: (b) Status of Running and Retired Jobs



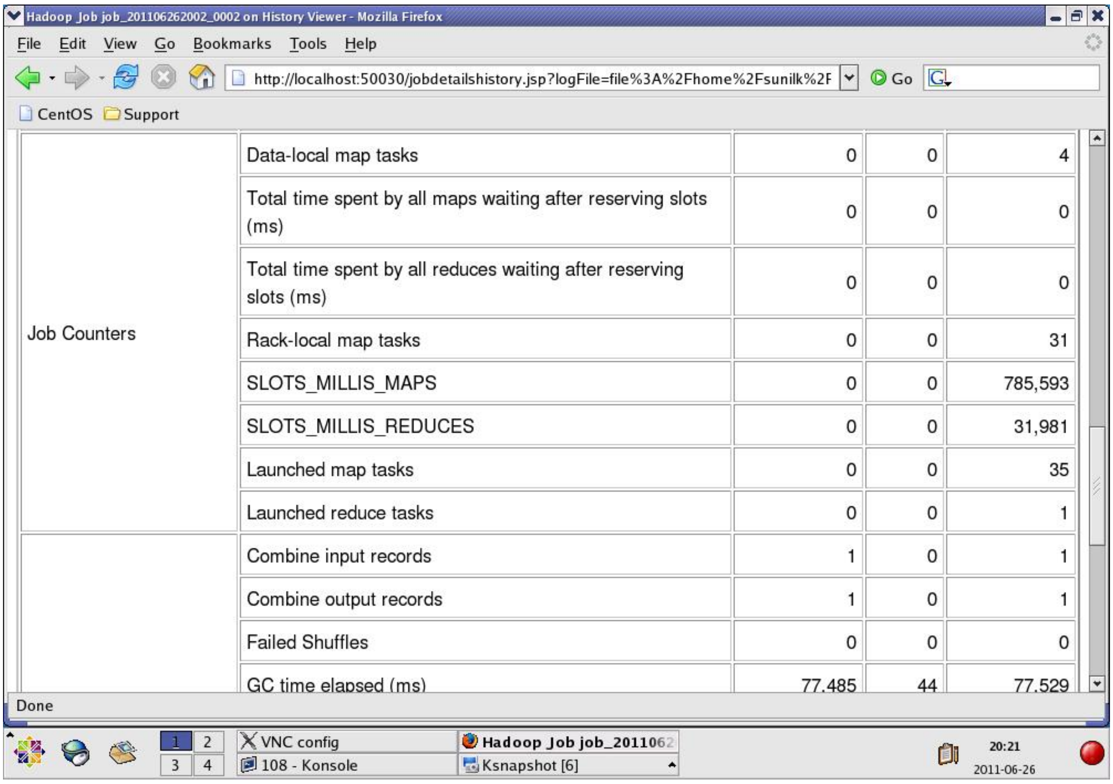
Kind	Total Tasks(successful+failed+killed)	Successful tasks	Failed tasks	Killed tasks	Start Time	Finish Time
Setup	1	1	0	0	26-Jun-2011 19:23:57	26-Jun-2011 19:23:58 (1sec)
Map	35	35	0	0	26-Jun-2011 19:22:00	26-Jun-2011 20:15:47 (53mins, 46sec)
Reduce	1	1	0	0	26-Jun-2011	26-Jun-2011 20:15:58

Figure 5.4: Status of Map/Reduce Successful, Failed, Killed tasks and start and Finish Time



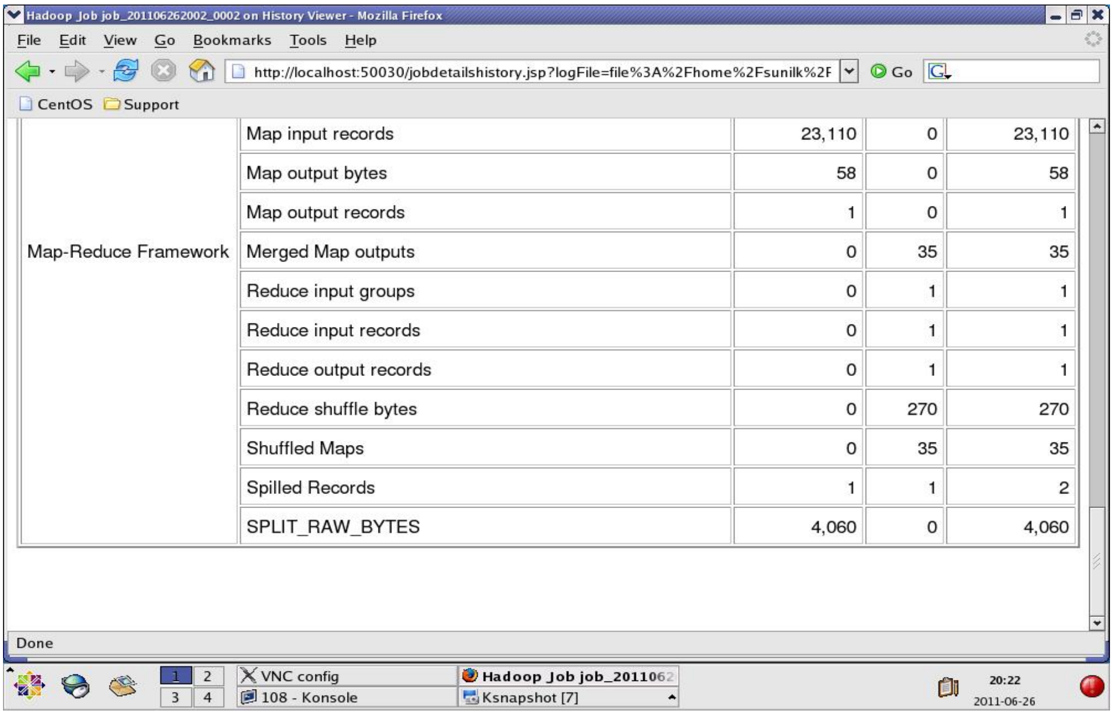
	Counter	Map	Reduce	Total
FileInputFormatCounters	BYTES_READ	2,342,864,050	0	2,342,864,050
FileSystemCounters	FILE_BYTES_READ	0	66	66
	FILE_BYTES_WRITTEN	1,390	66	1,456
	HDFS_BYTES_READ	2,344,682,638	0	2,344,682,638
	HDFS_BYTES_WRITTEN	0	61	61
Shuffle Errors	BAD_ID	0	0	0
	CONNECTION	0	0	0
	IO_ERROR	0	0	0
	WRONG_LENGTH	0	0	0
	WRONG_MAP	0	0	0
	WRONG_REDUCE	0	0	0
	Data-local map tasks	0	0	4

Figure 5.5: (a) FileInputFormatCounters, FilesSystemCounters,Shuffle Errors Parameter values on Mapper and Reducer contd.



Job Counters	Data-local map tasks	0	0	4
	Total time spent by all maps waiting after reserving slots (ms)	0	0	0
	Total time spent by all reduces waiting after reserving slots (ms)	0	0	0
	Rack-local map tasks	0	0	31
	SLOTS_MILLIS_MAPS	0	0	785,593
	SLOTS_MILLIS_REDUCES	0	0	31,981
	Launched map tasks	0	0	35
	Launched reduce tasks	0	0	1
	Combine input records	1	0	1
	Combine output records	1	0	1
	Failed Shuffles	0	0	0
	GC time elapsed (ms)	77,485	44	77,529

Figure 5.5: (b) Job Counters Parameter values on Mapper and Reducer



Map-Reduce Framework	Map input records	23,110	0	23,110
	Map output bytes	58	0	58
	Map output records	1	0	1
	Merged Map outputs	0	35	35
	Reduce input groups	0	1	1
	Reduce input records	0	1	1
	Reduce output records	0	1	1
	Reduce shuffle bytes	0	270	270
	Shuffled Maps	0	35	35
	Spilled Records	1	1	2
	SPLIT_RAW_BYTES	4,060	0	4,060

Figure 5.5: (c) Map-Reduce Framework Parameter values on Mapper and Reducer

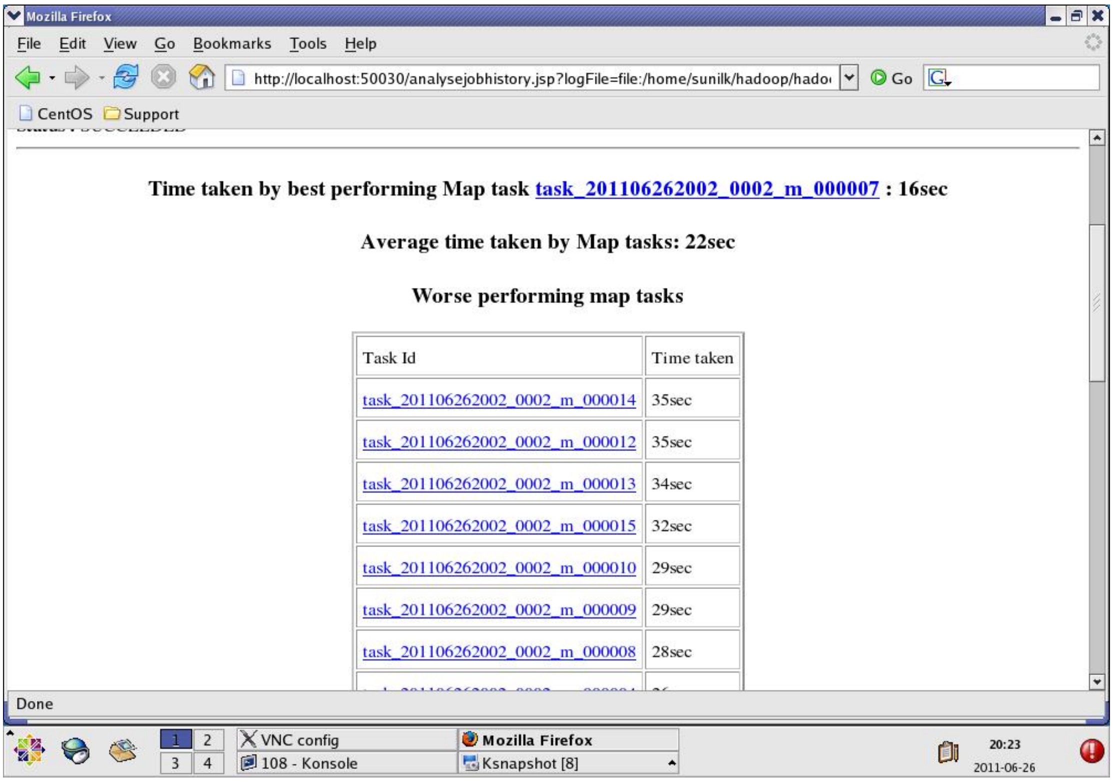


Figure 5.6: (a) Best performing Map Task and Time taken by Different Map Tasks

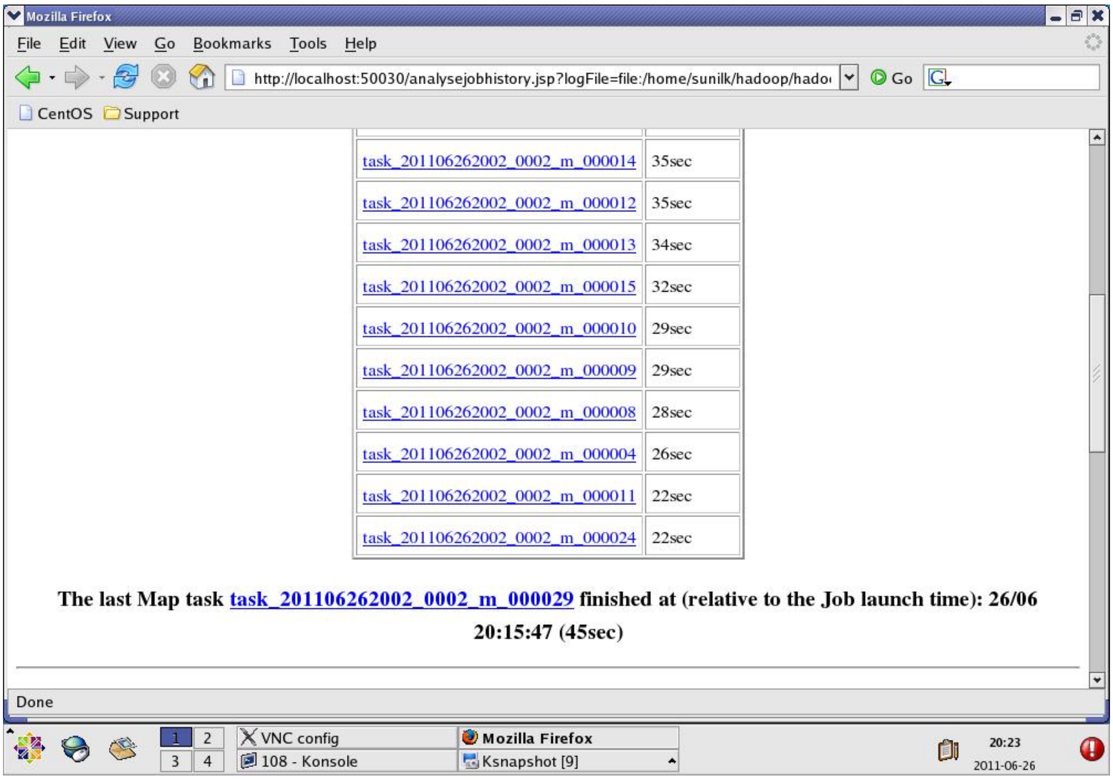


Figure 5.6: (b) Time taken by Different Map tasks contd. and Finishing Time of Last Map Task

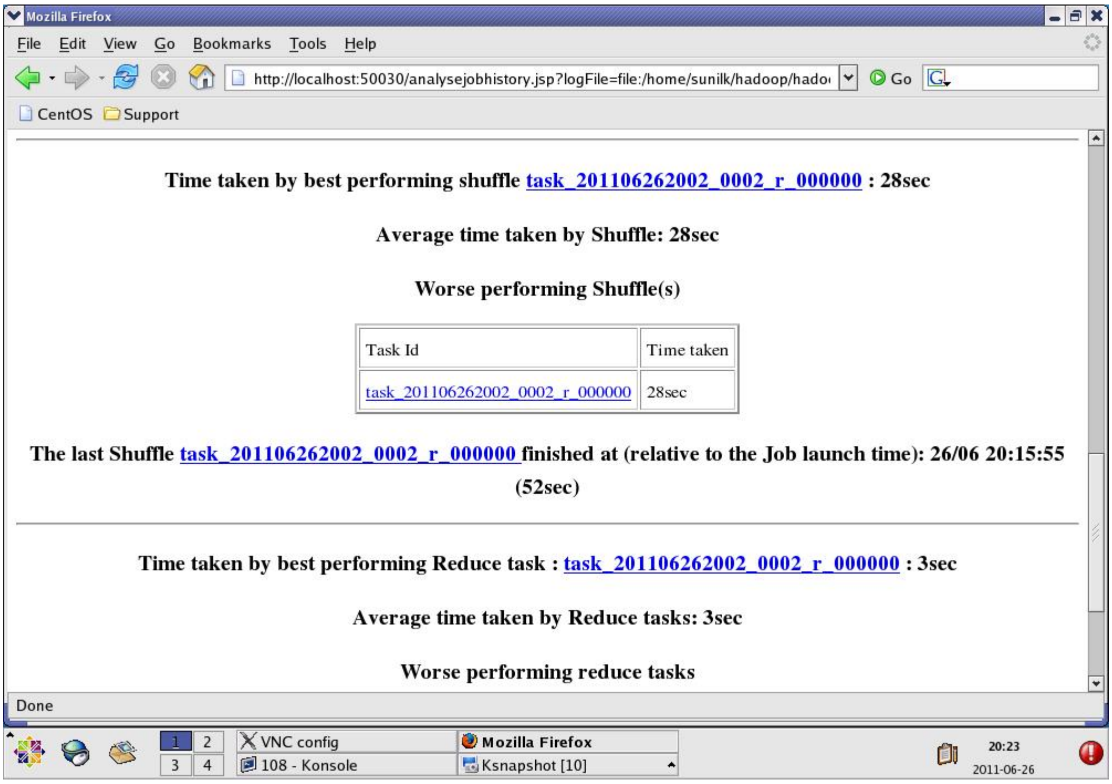


Figure 5.7: (a) Best, Average and Worst Time of shuffle.

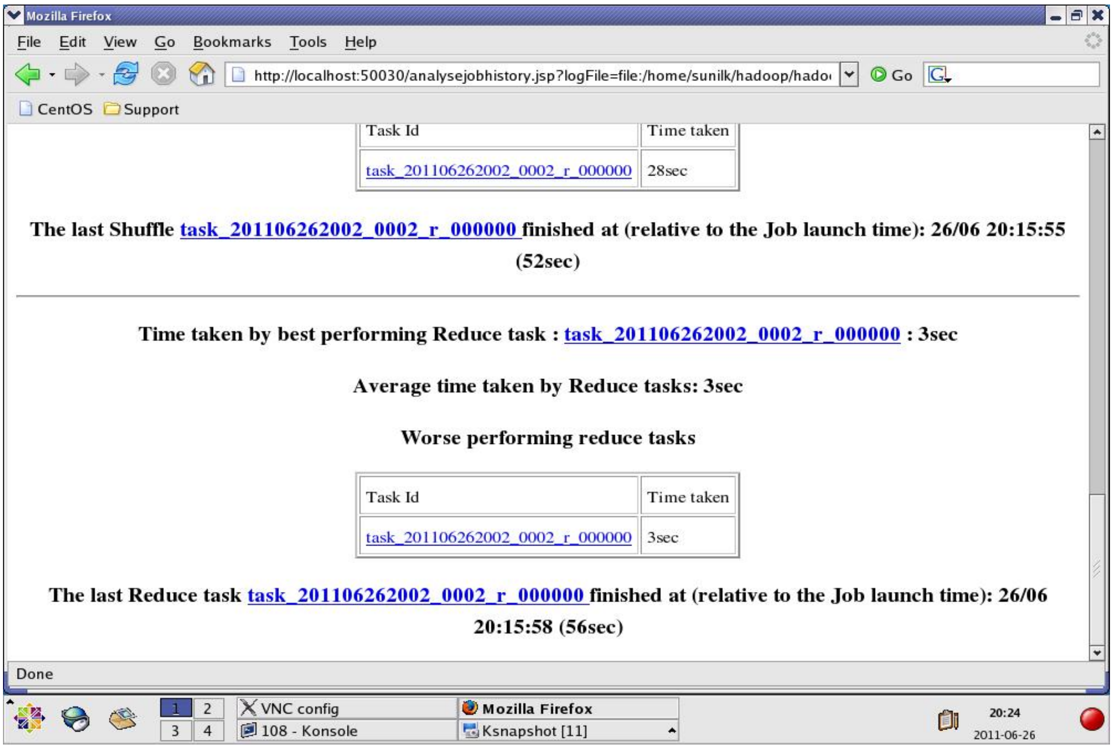
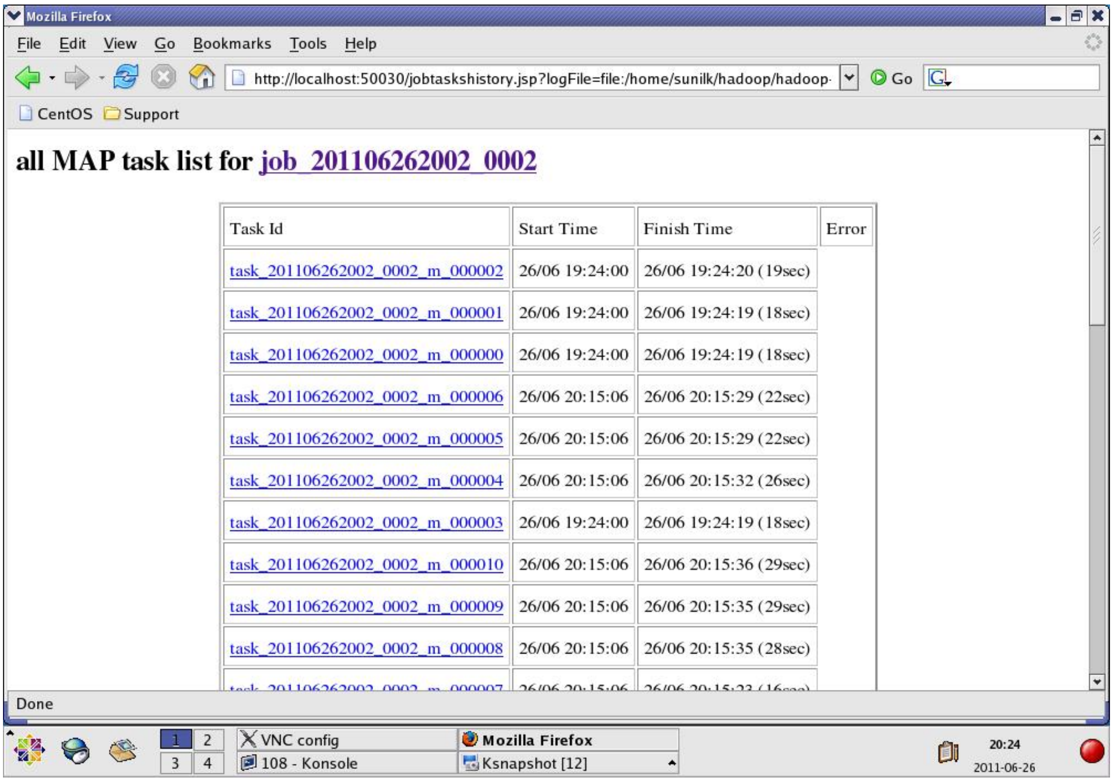


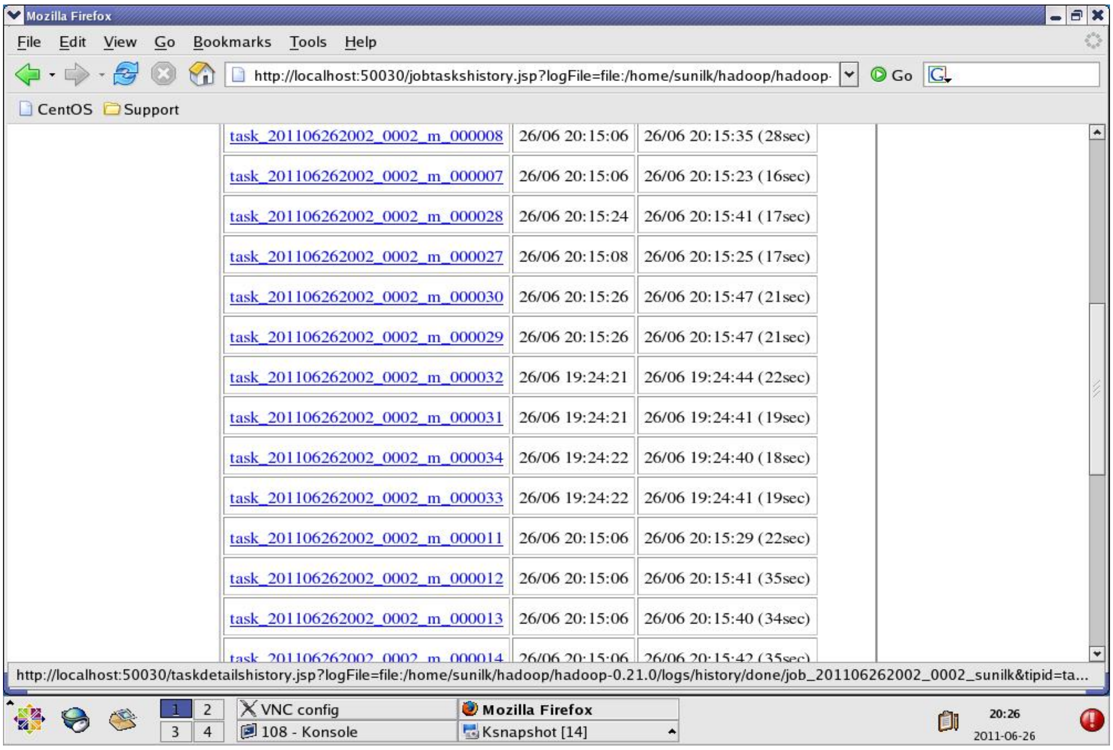
Figure 5.7: (b) Best, Average and Worst Time of Reducer.



all MAP task list for job_201106262002_0002

Task Id	Start Time	Finish Time	Error
task_201106262002_0002_m_000002	26/06 19:24:00	26/06 19:24:20 (19sec)	
task_201106262002_0002_m_000001	26/06 19:24:00	26/06 19:24:19 (18sec)	
task_201106262002_0002_m_000000	26/06 19:24:00	26/06 19:24:19 (18sec)	
task_201106262002_0002_m_000006	26/06 20:15:06	26/06 20:15:29 (22sec)	
task_201106262002_0002_m_000005	26/06 20:15:06	26/06 20:15:29 (22sec)	
task_201106262002_0002_m_000004	26/06 20:15:06	26/06 20:15:32 (26sec)	
task_201106262002_0002_m_000003	26/06 19:24:00	26/06 19:24:19 (18sec)	
task_201106262002_0002_m_000010	26/06 20:15:06	26/06 20:15:36 (29sec)	
task_201106262002_0002_m_000009	26/06 20:15:06	26/06 20:15:35 (29sec)	
task_201106262002_0002_m_000008	26/06 20:15:06	26/06 20:15:35 (28sec)	

Figure 5.8(a) Start Time and Finish Time of Different Map Tasks on Different Machines



task_201106262002_0002_m_000008	26/06 20:15:06	26/06 20:15:35 (28sec)	
task_201106262002_0002_m_000007	26/06 20:15:06	26/06 20:15:23 (16sec)	
task_201106262002_0002_m_000028	26/06 20:15:24	26/06 20:15:41 (17sec)	
task_201106262002_0002_m_000027	26/06 20:15:08	26/06 20:15:25 (17sec)	
task_201106262002_0002_m_000030	26/06 20:15:26	26/06 20:15:47 (21sec)	
task_201106262002_0002_m_000029	26/06 20:15:26	26/06 20:15:47 (21sec)	
task_201106262002_0002_m_000032	26/06 19:24:21	26/06 19:24:44 (22sec)	
task_201106262002_0002_m_000031	26/06 19:24:21	26/06 19:24:41 (19sec)	
task_201106262002_0002_m_000034	26/06 19:24:22	26/06 19:24:40 (18sec)	
task_201106262002_0002_m_000033	26/06 19:24:22	26/06 19:24:41 (19sec)	
task_201106262002_0002_m_000011	26/06 20:15:06	26/06 20:15:29 (22sec)	
task_201106262002_0002_m_000012	26/06 20:15:06	26/06 20:15:41 (35sec)	
task_201106262002_0002_m_000013	26/06 20:15:06	26/06 20:15:40 (34sec)	

Figure 5.8(b) Start Time and Finish Time of Different Map Tasks on Different Machines

Conclusion and Future Work

The need for secure methods of authentication is becoming increasingly important in the corporate world today. Passwords, token cards and PINs are all risks to the security of an organization due to human nature. Our inability to remember complex passwords and tendency to write these down along with losing token cards or forgetting PIN number all contribute to the possible breakdown in security for an organization.

Main objective of proposed work is to provide a framework for biometrics systems on cloud computing. Results shows how effectively cloud computing can be used in biometrics systems where data set size is growing very fast. By increasing the number of compute nodes we can achieve same speedup and efficiency gain. Cloud computing also gives the scope for use of biometrics systems in security system (Airport, Home Land security, Department of Defense). It is also observed that while Hadoop framework is highly scalable, its scalability also has to depend on size of dataset.

The future scope of work is to deploy multimodal biometrics system on cloud computing, where data set size and computation time will be much large. On Hadoop framework, dynamically machines can be added as data set size grows.

References

[1] http://www.technicalsymposium.com/Iris_Recognition.html

[2] Wikipedia web site: [online available] http://en.wikipedia.org/wiki/Iris_recognition

[3] 10 reasons Why Cloud Computing is the Wave of the Future, April 29, 2009, by Gilberto J. Perera [online]
URL: http://laptoplogic.com/resources/10-reasons-why-cloud-computing-is-the-wave-of-the-future

[4] SLIC: Short-Length Iris Codes, James E. Gentile, Nalini Ratha and Jonathan Connell IBM T.J. Watson Research Center

[5] Wikipedia on cloud computing [online]
URL: http://en.wikipedia.org/wiki/Cloud_computing

[6] http://www.ibm.com/developerworks/aix/library/au-cloud_apache/

[7] T. White, Hadoop: The Definitive Guide. O'Reilly Media, [online]:
URL: http://oreilly.com/catalog/0636920010388/preview

[8] Apache Hadoop HDFS Architecture Guide. [Online]
URL: http://hadoop.apache.org/hdfs/docs/current/hdfs_design.html

[9] A Method for Iris Recognition Based on 1D Coiflet Wavelet Agus Harjoko, Sri Hartati, and Henry Dwiyasa

[10] Online available: http://scgwww.epfl.ch/courses/Biometrics-Lectures-2010-2011-pdf

[11] “Don’t Blink: Iris Recognition for Biometric Identification ”, Mary Dunker , SANS Security Essentials, July 2003 , GSEC Certification Practical, Version 1.4b

[12] S. Sanderson, J. Erbetta. “Authentication for secure environments based on iris scanning technology”. IEE Colloquium on Visual Biometrics, 2000.

[13] J. Daugman. “How iris recognition works”. Proceedings of 2002 International Conference on Image Processing, Vol. 1, 2002.

[14] E. Wolff. Anatomy of the Eye and Orbit. 7th edition. H. K. Lewis & Co. LTD, 1976.

[15] Kevin W. Bowyer, Karen Hollingsworth, Patrick J. Flynn, “Image Understanding for Iris Biometrics: A Survey, Computer Vision and Image Understanding”, Vol. 110, Issue 2, pp. 281-307, 2008.

[16] W. Ketchantang, S. Derrode, S. Bourennane and L. Martin, “Video Pupil Tracking for Iris Based Identification”, Advanced Concepts for Intelligent Vision Systems, LNCS 3708, pp. 1-8, 2005.

[17] R. Wildes, J. Asmuth, G. Green, S. Hsu, R. Kolczynski, J. Matey, S. McBride. “ A system for automated iris recognition.” Proceedings IEEE Workshop on Applications of Computer Vision, Sarasota, FL, pp. 121-128, 1994.

[18] C. Tisse, L. Martin, L. Torres, M. Robert. “Person identification technique using human iris recognition.” International Conference on Vision Interface, Canada, 2002.

[19] W. Kong, D. Zhang. “Accurate iris segmentation based on novel reflection and eyelash detection model.” Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing, Hong Kong, 2001.

[20] L. Ma, Y. Wang, T. Tan. “Iris recognition using circular symmetric filters.” National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, 2002.

[21] The Hadoop Distributed File System: Architecture and Design [online available]

http://clustercenter.org/storage/Hadoop-Distributed-File-System-Architecture-Design

[22] Distributed computing with Linux and Hadoop [online available]

http://www.ibm.com/developerworks/linux/library/l-hadoop/

[23] Cloud computing with Linux [online available]

http://www.ibm.com/developerworks/library/l-cloud-computing/index.html

[24] IBM MapReduce Tools for Eclipse [online available]

http://www.alphaworks.ibm.com/tech/mapreducetools

[25] Hadoop Map/Reduce, Owen O'Malley, July 2006 [online available]

http:///apache/hadoop/HadoopMapReduceArch.pdf

[26] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler, "The Hadoop Distributed File System." ISBN 978-1-4244-7153-9.

[27] Hadoop Distributed File System, Dhruba Borthakur, June, 2007 [online available] http://apache/hadoop/HDFSDescription.pdf

[28] Biometric Identification on Cloud Computing

http://www.afisandbiometrics.com/biometric_identification_on_cloud_computing

[29] Booz Allen Ideas Festival web site [online]. Available: http://www.boozallen.com/insights/bozo-allen-ideas-festival/winning-ideas/ideas-cloud-biometrics

[30] Padma Polash, P. Maruf Monwar, M. Ahsanullah," Human iris recognition for biometric identification" Computer and information technology, 2007. iccit 2007. 27-29 Dec. 2007, ISBN: 978-1-4244-1550-2,pp 1-5.

[31] IITD Iris Database, web site available: http://www4.comp.polyu.edu.hk/~csajaykr/IITD/Database_Iris.html

[32] MMU iris Database, web site available: http://pesona.mmu.edu.my/~ccteo/

[33] UBIRIS Database web site available: http://iris.di.ubi.pt/

[34] CASIA iris Database, web site available: http://www.cbsr.ia.ac.cn/english/IrisDatabase.asp

[35] Yu Shyang Tan, Bu-sung Lee, “MapReduce Framework: Some Challenges”, Annual International Conference on CCV 2010, pp. 278-284

[36] Mohamed, M.A. Abou-El-Soud, M.A. Eid, M.M. Fac. of Eng., Mansoura Univ., Mansoura, Egypt, “Automated algorithm for iris detection and code generation”, Computer Engineering & Systems, 2009. ICCES 2009. 14-16 Dec. 2009 **ISBN:** 978-1-4244-5842-4, pp 475 – 481.