I.J. Modern Education and Computer Science, **2019, 12, 47-52**
Published Online December 2019 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2019.12.05

Analysis of Large Set of Images Using MapReduce Framework

Sawsan M. Mahmoud
Mustansiriyah University/College of Engineering, Computer Engineering Department, Baghdad, Iraq
Email: sawsan.mahmoud@uomustansiriyah.edu.iq

Rokaia Shalal Habeeb
Mustansiriyah University/College of Engineering, Computer Engineering Department, Baghdad, Iraq
Email: rokaia.shalal @uomustansiriyah.edu.iq

Received: 11 October 2019; Accepted: 28 October 2019; Published: 08 December 2019

Abstract—Due to the limitations of a physical memory, it is quite difficult to analyze and process big datasets. The Hadoop MapReduce algorithm has been widely used to process and mine such large sets of data using the Map and Reduce functions. The main contribution of this paper is to implement MapReduce programming algorithm to analyze large set of fingerprint images which cannot be normally processed due to a limited physical memory in order to find the features of these images at once. At first, the images are maintained in an image data store in order to be preprocessed and to extract the features for the biometric trait of each user, and then store them in a database. The algorithm preprocesses and extracts the features (ridges and bifurcation) from multiple fingerprint images at the same time. The extracted points are detected using the Crossing Number (CN) concept based on the proposed algorithm. It is validated using data taken from the National Institute of Standards and Technology's (NIST) Special Database 4. The data consist of fingerprint images for many users. Our experiments on these large set of fingerprint images shows a significant reducing in the processing time to a nearly half when extracting the features of these images using our proposed MapReduce approach.
.

Index Terms—MapReduce Programming, Fingerprint Image, Feature Extraction, Minutiae Extraction, Crosssing Number Algorithm.

I. Introduction

In recent years, large sets of images have been stored in many social media sites. Processing these huge data resources may lead to bottlenecks due to single computers, power, and how much storage is needed. Alternately, in distributed systems, the tasks typically are performed by dividing them into various subtasks. Normally, when the tasks are parallelized, the resource-intensive applications are more scalable and efficiently executed. A good platform for such tasks is provided by the Hadoop MapReduce algorithm [1].

There has been an expansive amount of research dedicated to image processing along with the Hadoop MapReduce algorithm [2] [3]. For example, in [4], a privacy-preserving and content-based system is proposed to search large sets of images. In that work, cloud and client side is included in order to implement the system. The cloud is represented by a cluster of computers that has the distributed file system Hadoop -HDFS and the MapReduce framework Hadoop–MapReduce. The system is evaluated using real life pictures.

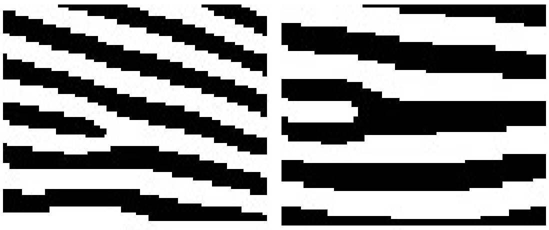
XHAMI an extended MapReduce and HDFS interface on an application of image processing, is implemented in [5]. The authors used XHAMI as an extended library of both HDFS and MapReduce, which are used to read and write the single largescale images. In addition, in [1], Hadoop for image processing context is proposed. In this way, the Hadoop system implements the image processing without needing the particular expertise of the programmer in distributed systems since the details of the Hadoop system are hidden. The authors in [6] suggest a scalable system to read large volumes of images using the Hadoop HDFS and the MapReduce algorithm. The system has been tried to explain a merging process with reference to finding the most similarities between two images.

In this paper, the Hadoop MapReduce algorithm is used for minutiae feature extraction from a large set of fingerprint images stored in a data store. Two functions are used to deal with these large set of fingerprint images: mapper and reducer functions. The map function finds the morphological thinning, ridges ending, and bifurcation while the reducer function aggregates them for extracting the features of a large number of images in parallel. In most of the current fingerprint matching systems, the features used in the matching process are the fingerprint minutiae, mainly ridges bifurcation and ridges ending. By using MapReduce programming, the fingerprint minutiae are obtained in parallel to identify the features.

The rest of the paper is organized as follows: Section two presents an overview of the fingerprint feature extraction in image processing. Section three introduces the Hadoop MapReduce framework as well as a description of distributing tasks for MapReduce. In Section four, the simulation results of applying the MapReduce function on fingerprint images to extract their features are presented. Finally, conclusions are introduced in section five.

II. Feature Extraction

Biometric identification refers to recognizing individuals by anatomical (e.g., iris, fingerprints, face) and behavioral characteristics. Fingerprints are one of the oldest biometric identifiers used in biometric identification [7]. They are almost always unique and unalterable during the lifetime of an individual. The trace of valleys and ridges that contours the skin surface of a finger form the fingerprint. Ridges are usually curved lines that contain valleys in-between them [8]. Other terms used in fingerprint recognition, which are dominant as an identity marker for an individual, are the ridge ending and bifurcation or simply (minutiae). The termination point of a ridge line is called a ridge ending, while a Y-shape split of the ridge line forms the bifurcation. The most commonly used features in fingerprints are the minutiae [9,10,11]. An example of a bifurcation and a ridge ending are shown in Fig. 1. The ridges are represented in the black areas, while the valleys are represented in the white areas [7]. A feature extraction stage is required for most of the fingerprint recognition and classification algorithms [7].



(a) Ridge Ending (b) Bifurcation

Fig.1. Types of minutiae

A. *Minutiae Extraction Techniques*

Minutiae matching is used in most automatic recognition systems as a base for fingerprint comparison, therefore, minutiae extraction is an essential stage in these systems [7]. Generally, minutiae extraction techniques can be categorized into two main methods as shown in Fig. 2.

- Methods that use grayscale fingerprint images.
- Methods that use binarized fingerprint images [12].

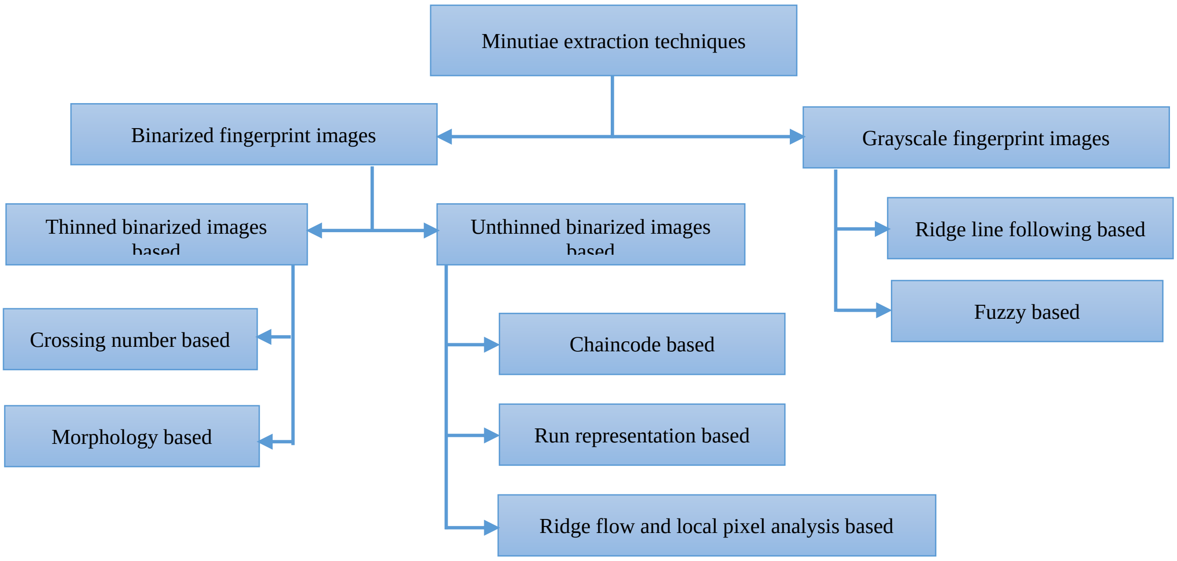

Minutiae extraction techniques
Binarized fingerprint images
Grayscale fingerprint images
Thinned binarized images based
Unthinned binarized images based
Ridge line following based
Fuzzy based
Crossing number based
Morphology based
Chaincode based
Run representation based
Ridge flow and local pixel analysis based

Fig. 2. Minutiae Extraction Techniques [12]

In this paper, a technique based on binarized images is used to extract the fingerprint features. It is important to state that most of the proposed techniques in the literature require image enhancement before the binarization stage [7]. Here we use histogram equalization as an enhancement or preprocessing before the binarization stage.

B. *Minutiae Extraction Stages*

Generally, there are three main stages to extract the minutiae from a fingerprint image as explained in Fig. 3:

1) Binarization Stage

In this stage the gray-scale image of the fingerprint must be transformed into a binary image by comparing the gray level value of each pixel to a certain threshold. The pixels with a gray level value of less than a specific threshold are considered to be 0, while the rest of pixels are considered to be 1. Binarization facilitates minutiae extraction by improving the contrast between the ridges and valleys [12].

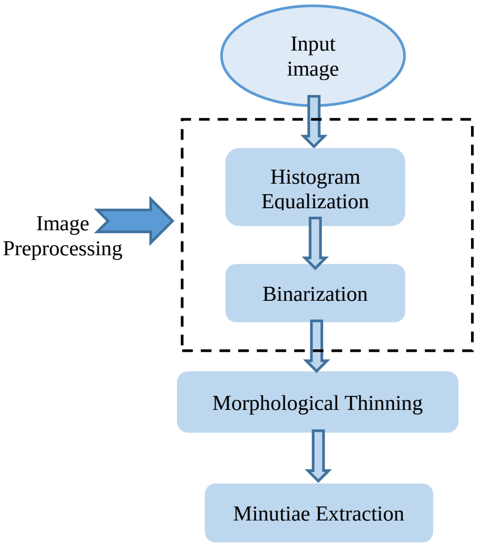
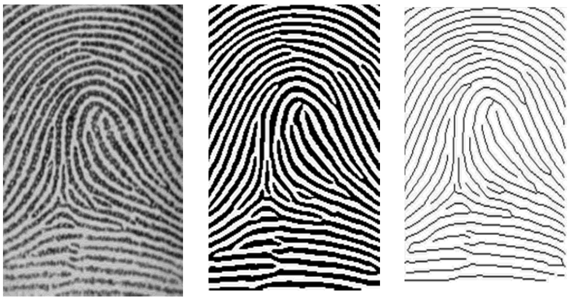


Fig.3. Minutiae Extraction Stages

2) Thinning Stage

The thickness of the ridge line in the binarized image is reduced to one pixel by applying the thinning stage. This process provides a skeleton-shape image [7]. An example of a fingerprint gray-scale image, a binarized image, and a skeleton image are shown in Fig. 4 [7]



(a) Fingerprint gray-scale image (b) Binary Image (c) Skeleton image

Fig. 4. Feature Extraction Stages

3) Minutiae Detection Stage

Ridge endings and bifurcations are extracted from a thinned skeleton fingerprint image. In this paper, the CN technique is adopted to extract minutiae, where the minutiae points are detected based on the number of neighboring pixels [12]. CN is widely used to extract the minutiae. It is applied to a binary skeleton fingerprint image. The CN can be calculated by summing the differences of adjacent pixels surrounding a certain foreground pixel Px, then dividing the result by two as shown in (1) [7]:

$$CN(P_x) = 0.5\sum_{i=1}^{8}|P_{xi} - P_{xi+1}| \qquad (1)$$

Where Px_i is the i^{th} binary pixel value in the neighborhood of Px with (i=1, 2, 3. . . 8) Px_i = (0 or1) and Px_9 = Px_1. The neighboring pixels of pixel Px are explored by a 3 × 3 window in an anti-clockwise direction as shown in Table 1:

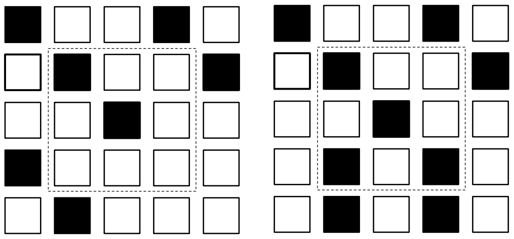
Table 1.Pixel Px and its Eight Adjacent Pixels

Px_4	Px_3	Px_2
Px_5	Px	Px_1
Px_6	Px_7	Px_8

The pixels are then categorized based on their CN value as shown in Table 2. Fig. 5 illustrates how a CN value can differentiate between a ridge ending and a bifurcation.

Table 2.CN Properties

CN	Property
0	Isolated point
1	Ridge Ending
2	Connective point
3	Bifurcation
4	Crossing point



(a) Ridge Ending (b) Bifurcation

Fig. 5. CN Differencing Between a Ridge Ending and a Bifurcation [7]

Finally these pixels or the orientation of the pixels are organized in a feature vector *F* (X_i, Y_i) which contains the index of each pixel in order to use it for matching or verification.

III. Feature Extraction Based on a MapReduce Framework

MapReduce is used as a programming technique for processing large sizes of data running on a distributed computing environment [13]. It is used for handling data that are insufficient to be stored in a physical memory [14]. A MapReduce algorithm consists of two functions: map and reduce programming functions. These two functions are performed in two steps that are separated by data transfer that is between nodes in a cluster. These steps are processed in parallel using data as {key, value} pairs. The map function execution step starts by taking a value from the input dataset as input and then implements the function to that value, thereby producing intermediate output results. These intermediate results are also in the form of {key, value} pairs of records kept in the nodes of the cluster. The records for any key could go through many nodes. The output from the map function is sorted in order to be input for the reduce function. This includes data transfers between the map and reduce functions. The reduce function execution step starts when all the data output from the map function step is transmitted to the suitable machine. As the nodes run the reduce function for a certain key, the values are aggregated at that node. The final output is produced by the reduce function in the form {key, value} as well [13].

MapReduce programming functions are essentially processed in parallel and hence, large volumes of data analysis are processed with enough machine when putting into the hands of anyone at their disposal. Generally, the MapReduce algorithm is very good at mining large volumes of datasets that are of a petabyte size and cannot be stored into a physical memory [13] [15].

In this paper, MapReduce functions are used to deal with large sets of images in order to find their features. The overall flow of the mapper and reducer functions are implemented as shown in Fig. 6. The map function finds the morphological thinning, ridges ending, and bifurcation while the reducer function aggregates them for extracting the features of a large number of images in parallel.

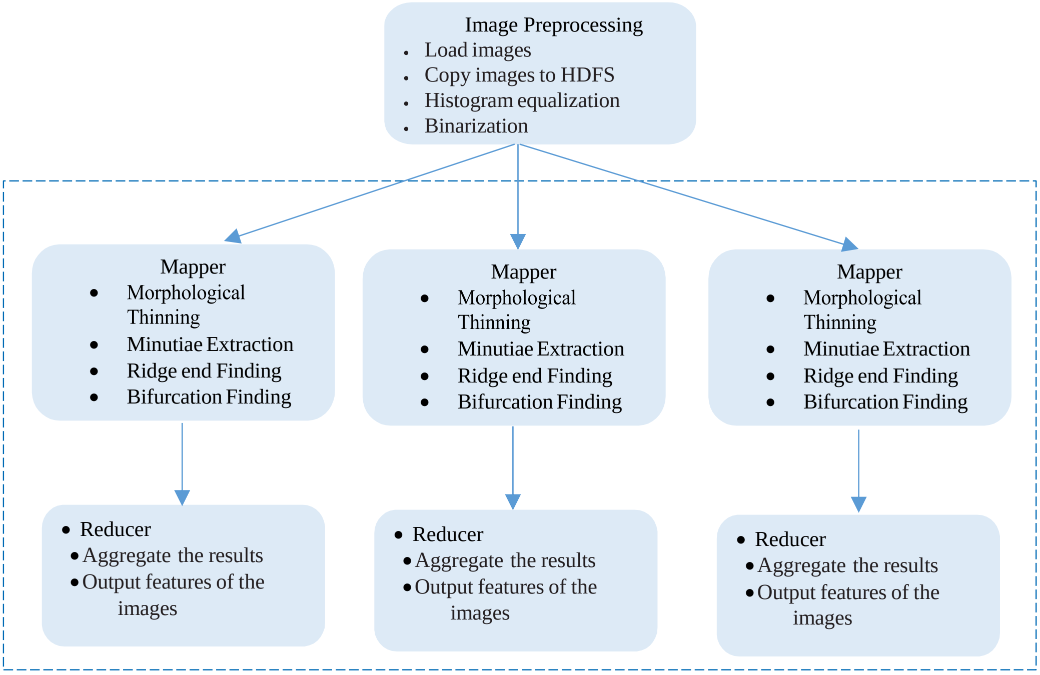


Fig. 6. Feature Extraction Based on a MapReduce Framework

IV. Simulation Results

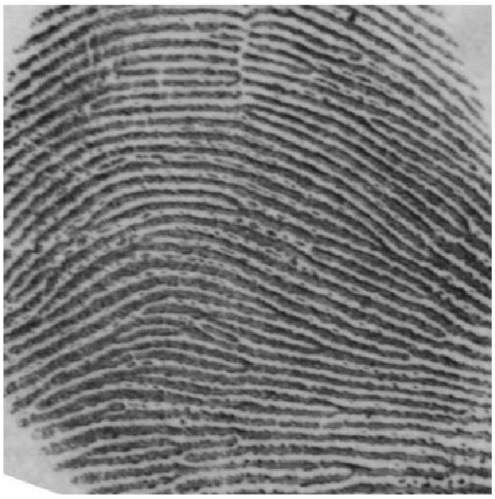
The Hadoop MapReduce framework system is implemented here to extract the features of a large set of fingerprint images. The experimental results are carried out on a Personal Computer where Windows 10 Pro is installed with a 2.60 GHz processor and 8 GB of installed memory (RAM). In addition, the parallel pool of Matlab (Matrix Laboratory) is used in order to execute Parallel MapReduce as the front- end engine. Through the experiments, the pool is connected to two workers.

1) Datasets: The data used in our experiments are different sets of images in which their features are extracted in parallel. These images are downloaded from the National Institute of Standards and Technology’s (NIST) Special Database 4 [16]. The database contains different 8-bit grayscale randomly selected fingerprint images. The database file has about 4000 (2000 pairs) fingerprint images in the PNG format. The database is being used for testing fingerprint classification systems. Each fingerprint consists of 512x512 pixels with 32 rows of white space located at the bottom of the fingerprint image.

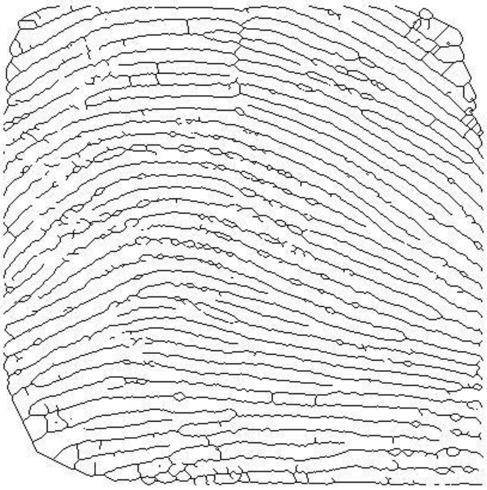
2) Fingerprint Features Extraction Results: A random sample of different fingerprint images from the NIST-4 dataset are selected and stored to perform the extraction algorithm based on the suggested MapReduce functions. MapReduce functions deal with a large collection of data sources. Hence, a greater cloud is required when working with such large collection of data. The images are stored in a data store in order to be processed later in parallel. Before extracting the features of the images, these images should be preprocessed. As an enhancement to the fingerprint image, histogram equalization is implemented for each image before the binarization stage. The binary images are sent to the next stage where a morphological thinning is applied that produces a skeletonized image of the fingerprint. Finally, minutiae are extracted from the skeleton images using the CN technique resulting in a feature vector that contains the indices of the ridge endings and bifurcations in the image. As stated previously, the Hadoop MapReduce algorithm consists of two functions:

Mapper and Reducer. In the mapper functions, the images are loaded into the memory, and the steps of feature extraction are then applied to the set of images stored on the HDFS. Accordingly, the thinning and minutiae extraction stages are implemented during the Hadoop Map step. The morphological thinning and minutiae extraction is done in parallel for all images in this step. Once all the mappers are executed, intermediate results are produced and passed to the reducers, where the results are combined in this stage of the MapReduce algorithm.

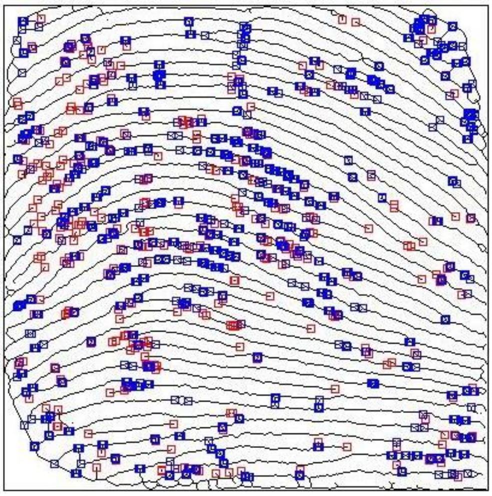
For example, Fig. 7 shows the features of one fingerprint image randomly chosen from the NIST-4 database. The processing time to find these features is 0.89748 seconds. It is important to state that this processing time (as calculated using tictoc Matlab command) is influenced by the hardware settings.



(a) Input Image



(b)Thinned Image



(c) Minutiae Extracted

Fig. 7. Feature extraction of Image f0015-01r from the NIST-4 data set

Although the feature extraction step takes about one second for a 512x512 fingerprint image, the total time taken to find the features for a large set of images can be very high since these features must be extracted for each fingerprint in the database. Alternatively, for instance, it has taken 26.282803 seconds to find the minutiae for 50 fingerprint images using the MapReduce functions. Hence, this reduces the processing time to nearly half. Fig. 8 shows the total processing time of feature extraction for different number of images, when processed in parallel using MapReduce framework.

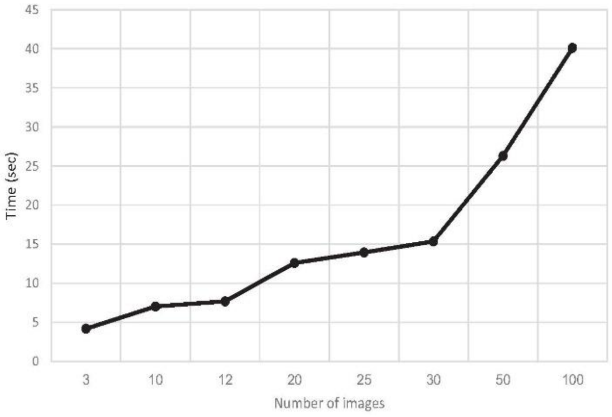


Fig. 8. Total Processing Time of Images Using Mapreduce

While Fig. 9 shows the relation between the average processing time per image, and the number of images processed in parallel using the suggested algorithm. It is noticed here that the average processing time per image to produce a feature vector for 3 images is about 1.4 seconds which takes more processing time than processing one image without using MapReduce algorithm. On the other hand this processing time is reduced as the number of images loaded to the mapper increased, so that the average time per image is about 0.4 second when processing 100 images..

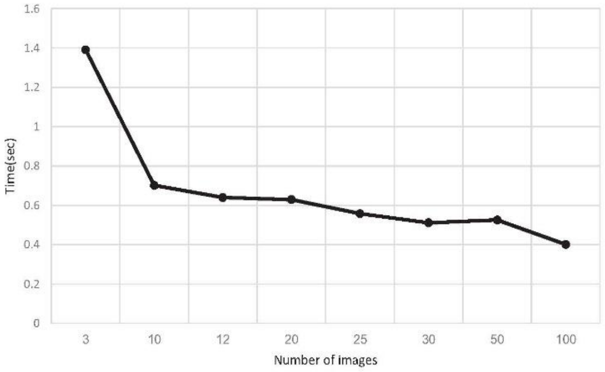


Fig. 9. Average Processing Time per Image Using Mapreduce.

V. Conclusions

In this study, the Hadoop MapReduce framework system is employed to find the features of large set of fingerprint images. The results show a substantial reduce in processing time per image to produce a feature vector. The gain in time when implementing MapReduce to extract the features of multiple images at once could be useful in forensic applications that require processing large numbers of fingerprint images.

References

[1] S. Vemula and C. Crick, “Hadoop image processing framework”, *IEEE International Congress on Big Data*, pp.506-513, 2015.

[2] Zhen Zhang, Wei Li, Hai Tao Jia, “A fast Face Recognition Algorithm based on MapReduce”, *Seven International Symposium on Computational Intelligence and Design*, 2014.

[3] Mohammed H. Almeer, “Cloud Hadoop MapReduce for Remote Sensing Image Analysis”, *Journal of Emerging Trends in Computing and Information Sciences*, Vol. 3, No. 4, pp. 637-644, April 2012.

[4] Lan Zhang, Taeho Jung, Puchun Feng, Xiang-Yang Li, Yun- hao Liu, “Cloud-based Privacy Preserving Image Storage, Sharing and Search”, arXiv:1410.6593[cs.CR]

[5] Raghavendra Kune, Pramod Kumar Konugurthi , Arun Agarwal, Raghavendra Rao Chillarige, and Rajkumar Buyya, “XHAMI - Extended HDFS and MapReduce Interface for Big Data Image Processing Applications in cloud computing environments” *Software: Practice and Experience* vol.47 pp. 455–472, 2017.

[6] H. Sarı, S. Eken, A. Sayar, “An Approach for Stitching Satellite Images in A Big data MapReduce Framework”, *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Volume IV-4/W4, 4th International GeoAdvances Workshop, 14–15 October 2017, Safranbolu, Karabuk, Turkey, 2017.

[7] Maltoni, D., Maio, D., Jain, A. K., and Prabhakar, S., “*Handbook of Fingerprint Recognition*”. Springer, 2nd edition 2009.

[8] Jain, A., Hong, L., Pankanti, S., and Bolle, R. “An identity authentication system using fingerprints*”. In Proceedings of the IEEE* (September 1997), vol. 85, pp. 1365–1388. 1010–1025, Aug. 2002.

[9] Gaensslen R. E., Ramotowski R., Lee H. C., “*Advances in Fingerprint Technology*”, 2nd edition CRC press, 2001.

[10] Leonard, B., “*Science of fingerprints: Classification and uses*”. Diane Publishing Co., ISBN:0-16-050541-0, Darby, Pennsylvania, 1988.

[11] Sheng W.et al. “A Memetic Fingerprint Matching Algorithm”, *IEEE Transactions on Information Forensics and Security*, Vol. 2, No. 3, September 2007, pp. 402-412.

[12] Bansal, Roli and Sehgal, Priti and Bedi, Punam, “Minutiae extraction from fingerprint images-a review”, *IJCSI International Journal of Computer Science Issues*, Vol. 8, Issue 5, September 2011.

[13] Prajesh P Anchalia, “Improved MapReduce k-Means Clustering Algorithm with Combiner”, *16th International Conference on Computer Modelling and Simulation*, UKSim, 2014.

[14] D. Willingham, “Big Data Analysis and Analytics with Matlab”, *Proceedings of ICALEPCS*, 2015.

[15] Tom White, “*Hadoop: The Definitive Guide*”, 4th edition, O’Reilly, 2015.

[16] National Institute of Standards and Technology, “*https://www.nist.gov/srd/nist−special−database−4*”, 2018.

Authors’ Profiles

Sawsan M. Mahmoud received her B.Sc. in Computer Science from University of Technology/Baghdad, Iraq in 1994. She obtained her M.Sc. from University of Baghdad in 1998. Her Ph.D. degree in Computational Intelligence is obtained from Nottingham Trent University, Nottingham, UK in 2012. Sawsan joined Mustansiriyah University/ Engineering College in 1994 as a member of the academic staff. Her research interests include, but not limited to, Computational Intelligence, Ambient Intelligence (Smart Home and Intelligent Environment), Wireless Sensor Network, Data Mining, and Health Monitoring.

Rokaia Shalal Habeeb is a lecturer at the computer engineering department, College of Engineering, Mustansiriyah University, Baghdad, Iraq since 1997. She received her B.Sc. in Electrical Engineering from Mustansiriyah University/Baghdad, Iraq in 1995 and her M.Sc. degree in Electronic and Communication Engineering from Mustansiriyah University/Baghdad in 2002. Her research interests include Intelligent Systems, Optimization Algorithms, and Image Processing.

How to cite this paper: Sawsan M. Mahmoud, Rokaia Shalal Habeeb, " Analysis of Large Set of Images Using MapReduce Framework", International Journal of Modern Education and Computer Science(IJMECS), Vol.11, No.12, pp. 47-52, 2019.DOI: 10.5815/ijmecs.2019.12.05