Implementation of Image Watermarking Processes on Cloud Computing Environments*

Chao-Tung Yang, Chu-Hsing Lin**, and Guey-Luen Chang

Department of Computer Science, Tunghai University, Taichung City 40704, Taiwan
{ctyang,chlin,g98357010}@thu.edu.tw

Abstract. With the faster Internet and data separation method were used widely and quickly, digital image watermarking becomes an important topic of intellectual property in the digital age. This paper proposes a method that can process image watermarking based on a robust method which combines the Singular Value Decomposition (SVD) and Distributed Discrete Wavelet Transformation (DDWT) over cloud computing environments. Hadoop system with the integrate functions, HDFS and MapReduce will play the key roles for this implementation.

Keywords: Hadoop, Watermarking, DDWT, SVD.

1 Introduction

Intellectual property of digital images has been a critical topic in the age of digitization. To enhance the security of digital images of large size such as those found in the National Digital Archives Program, we apply a watermarking technology [5-8-] based on DDWT [11][14] and SVD [1-2]. In order to efficiently process those digital images with large amount of data, we need high speed processors together with high throughput I/O.

DDWT with SVD as a novel digital watermarking method, in which robust watermarks is embedded into the cover image; and in this way, it can protect the ownership and also identify whether the source image is distorted. The intermediate frequency of DDWT and SVD are used to embed robust watermarks, and the spatial domain[3-5] is used to adaptively embed fragile watermarks. Results of simulation show that the proposed algorithm is very effective and can be widely used.

Hadoop, with reliable, scalable, and distributed computing characteristics, is open-source software and utilized heavily by Google. Here we take the advantage of these characteristics. HDFS[12], a distributed file system, provides high throughput access and low latency for data applications. And MapReduce[13], a software framework developed for distributed processing of vast amount of data sets on HDFS. In this paper, Hadoop is being used as a solution not only to provide the flexibility of

* This work is supported in part by the National Science Council, Taiwan R.O.C., under grants no. NSC 100-2622-E-029-008-CC3.

** Corresponding author.

R.-S. Chang, T.-h. Kim, and S.-L. Peng (Eds.): SUComS 2011, CCIS 223, pp. 131–140, 2011.

computing ability, but also to offer a way to solve the high performance computing that needs a high throughput I/O.

This rest of the paper is organized as follows. Section 2 introduces the background for our implementation requirement. Section 3 describes the implementation details of system architecture and computing criteria. Section 4 shows the results of the experiments; and finally, Section 5 gives the conclusions and the future work.

2 Background Review

2.1 SVD, DDWT and Combined Scheme

2.1.1 Singular Value Decomposition

Singular Value Decomposition (SVD) method is based on linear algebra. Chadra [1] proposed a new digital watermark scheme using singular value decomposition in 2002 to enhance the robustness of watermark against geometric and non-geometric attacks. It is also used in image compression [2-7], watermarking technologies [8-10], signal processing fields [3-5], noise estimation [9], etc.SVD is described as follows:

- The SVD embedding process

Step 1 Input original image X (M × N) and watermark image W (P × Q), do SVD to the original image X and the watermark W to obtain:

$$X=U^{x}\ \Sigma^{x}\ V_{x}^{T} \tag{1}$$

$$W=U^{w}\ \Sigma^{w}\ V_{w}^{T} \tag{2}$$

Where Σ_{x} and Σ_{w} means the singular value of the original image X and watermark W, respectively.

Step 2 Embed the singular value of watermark into the singular value of the original image.

$$\sigma_{y_i}=\sigma_{x_i}+(\sigma_{i}\times\sigma_{w_i}) \tag{3}$$

Step 3 The stego-image Y is obtained by

$$Y=U_{x}\ \Sigma_{Y}\ V_{x}^{T} \tag{4}$$

- The SVD extracting process

Step 1 Input the attacked image Y', uses SVD to obtain:

$$Y^{'}=U^{'}\ \Sigma^{'}_{Y}\ V^{'T} \tag{5}$$

Step 2 The singular matrix of extracted watermark is obtained as follows:

$$\Sigma^{'}_{w}=\frac{(\Sigma^{'}_{Y}-\Sigma^{'}_{X})}{\alpha} \tag{6}$$

Step 3 Multiply the three matrices to obtain the extracted watermark $W^{'}$

$$W^{'} = U_W \Sigma^{'}_W V_W \tag{7}$$

2.1.2 Multi-scale Distributed Discrete Wavelet Transformation (DDWT)

DDWT is based on Discrete Wavelet Transformation (DWT). As DWT is not resistant against cropping attacks, in 2006 Lin et a. [11][14] proposed the DDWT method. This approach use the multi-scaled DDWT to transform the image data from the space domain to the frequency domain, and then embed watermark information on the frequency domain. Process described as below.

- The DDWT embedding process

Step 1 Input original image $X_{M\times M}$ and watermark image $W_{N\times N}$;

Step 2 Using K-scale DDWT transform with X, where K is the number of scales and set scaling value t;

Step 3 Take the HL and LH from the K-scale DDWT transform and embed watermark into sub-band HL and LH using following equations:

The watermark extracting process

$$If\ W_{(i,j)} = 0; HL_{(i,j)} = t \times (2^k)^2 + HL_{(i,j)} \tag{8}$$

$$If\ W_{(i,j)} = 1; LH_{(i,j)} = t \times (2^k)^2 + LH_{(i,j)} \tag{9}$$

Step 4 Repeat step 3, until all of the watermark information were embedded;

Step 5 Do inverse DDWT to obtain the stego-image.

- The DDWT extracting process

Step 1 Input embedded image E and original image X ($M \times M$);

Step 2 Compute the block length l of image data for a single extracting process:

$$l = \left(\frac{M}{2^{k-1}} \right) \tag{10}$$

Step 3 Divide E and X with block length l into sub-blocks. Each image can be divided into s sub-blocks, $s = ((2^{k-1})^2)$ to obtain E_i and X_i, where $i \in \{1,2,3,\ldots s\}$;

Step 4 Subtract corresponding subsections from E and X, resulting in elements of array V_i, where $i \in \{1,2,3,\ldots s\}$;

Step 5 With each block of V, divide it into four square subsections with block length of (l/2). The sub-blocks are named as LL, HL, LH, and HH.

Step 6 Extract the individual pixel of the embedded watermark by equation

$$W_{(i,j)} = \begin{cases} 1, LL_{(i,j)} > 0 \quad and \quad LL_{(i,j)} > 0 \\ 0, LL_{(i,j)} > 0 \quad and \quad HL_{(i,j)} > 0 \end{cases} \tag{11}$$

Step 7 Repeat until all of the pixels in the embedded image are processed.

2.1.3 Watermarking Scheme Combined with DDWT and SVD

Our proposed watermark scheme combines the merits of SVD and DDWT [14]. SVD will provide the robustness against geometric attacks and non-geometric attacks. DDWT will provide the robustness against cropping attacks.

- **Water marking embedding process**:

Step 1 Input the original image $X_{M\times M}$ and the watermark $W_{N\times N}$

Step 2 Apply SVD on X and W:

$$X=U_x \Sigma_x V_x^T \tag{12}$$

Step 3 Eigenvalues embedding processing:

$$\sigma_y = \sigma_x + (\sigma \times \sigma_w) \tag{13}$$

Where σ_x are eigenvalues of Σ_x, σ_w are eigenvalues of Σ_w, σ_y are eigenvalues of Σ_y.

Step 4 Use SVD to obtain Y' :

$$Y'= U_x \Sigma_y V_x^T \tag{14}$$

Step 5 Process Y', in the 3-scale DDWT then embed watermarks into sub-bands LL3 and HH3

$$If\ W(i,j)=0\ then Y_{LL3}(i,j)=Y_{LL3}(i,j)+\alpha (2^k)^2; \tag{15}$$

$$If\ W(i,j)=1\ then Y_{HH3}(i,j)=Y_{HH3}(i,j)+\alpha (2^k)^2; \tag{16}$$

Step 6 Apply inverses DDWT to obtain the stego-image Y.

- **Watermark extracting process:**

Step 1 Imports the stego-image Y, the original image X, the image Y', and the watermark W.

Step 2 Subtract Y' from Y to obtain Y_{Diff}, and apply the following equation to extract the embedded watermark

$$W_{DDWT}(i,j)=\begin{cases}0, if\ Y_{DDWT}(i,j)<0\\ 1, \quad otherwise\end{cases} \tag{17}$$

Step 3 Apply SVD on X, Y' and W to find their eigenvalues σ_{Xi}, σ_{Wi}, σ_{Yi}.

$$Y= U_Y \Sigma_Y V_Y^T \tag{18}$$

$$W= U_w \Sigma_w V_w^T \tag{19}$$

$$X= U_x \Sigma_x V_x^T \tag{20}$$

Step 4 Extract Σ_{SVD} by using equation

$$\sigma_{SVD} = \frac{\sigma_{Yi} - \sigma_{Xi}}{\alpha} \tag{21}$$

Step 5 Apply SVD to obtain the SVD watermark W_{SVD} :

$$W_{SVD} = U_W \Sigma_{SVD} V_W^T \tag{22}$$

2.2 Hadoop Infrastructure

2.2.1 HDFS

Hadoop Distributed File System (HDFS) [12] is the storage system developed by apache project and used by Hadoop applications. It creates multiple replicas of data blocks, distributes them on compute nodes and throughout a cluster for reliable, extremely and rapid computations. HDFS is created for distributed storage and commodity hardware was designed to use for distributed processing. With the above characters, HDFS is simple to expand, fault tolerant, and scalability.

HDFS has master/slave architecture. HDFS cluster included a single namenode, a master server for management the file system, and regulates files access by clients. HDFS create a file system namespace and let user data could be stored. Internally, a file split into one or more blocks and stored in a set of datanodes. The namenode response for the namespace operations of the file system, for instance, opening, closing, renaming files, directories, and determines the mapping of blocks to datanodes. The datanodes are responsible for reading and writing requests from clients of the file system. Also, the datanodes in charge of the creation, deletion, and replication which indicate from the namenode.

2.2.2 MapReduce

MapReduce is an integral part and well known for its Simplicity, applicability, and process a large set of distributed applications of Hadoop. Hadoop MapReduce [11,13] process lot amounts of data with it characters in-parallel on large clusters of commodity hardware with a reliable, fault-tolerant method. MapReduce jobs spitted the input data-set into the independent chunks, these chunks are processed by the map tasks under a parallel method. The framework sorts the maps' output, then input to the next step, reduce tasks. The input and output files of the job will be stored in HDFS, then framework will takes care the scheduling tasks, monitoring the task and re-executes the failed tasks. The cluster of storage nodes are the same with datanodes, it's means that the MapReduce framework and the HDFS are running on the same set of cluster.

3 Implementation

In this section, we introduce our system architecture and how we composed those components. The Hadoop infrastructure plays a key role in entire system. The entire

system is according to official Hadoop manual. HDFS in charge of store the large data sets we need to process, offer the high throughput of data access rather than low latency. Mapreduce component in charge of the processes distributed. The master in charge of scheduling for the jobs' tasks on the slaves, and monitoring them. Our program specify (show in Figure 1) the input and output locations and supply map and reduce functions.

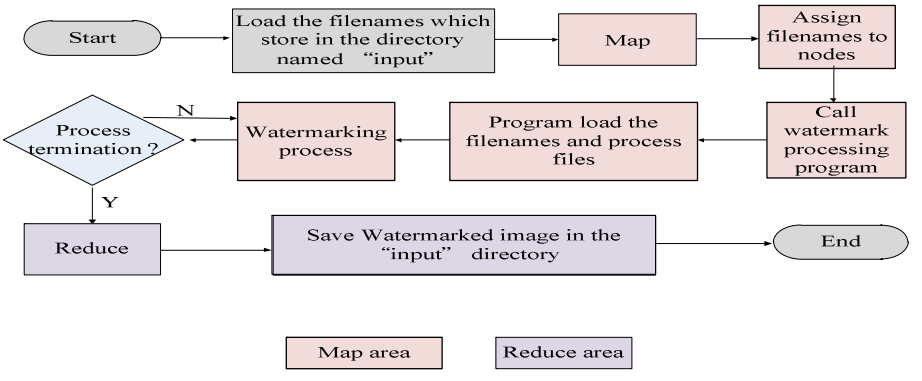

Fig. 1. Program flow

Table 1. Specification of hardware and software

Group	Node	Processor	Gflops	JAVA
Group 1	Node 1	Intel Xeon 2.0 G processor core x 4	1.22e+01	jre 1.6.0_24
	Node 2	Intel Xeon 2.0 G processor core x 4	1.23e+01	jre 1.6.0_24
	Node 3	Intel Xeon 2.0 G processor core x 4	1.22e+01	jre 1.6.0_24
	Node 4	Intel Xeon 2.0 G processor core x 4	1.24e+01	jre 1.6.0_24
Group 2	Node 5	Intel Xeon 3.2 G processor core x 2	6.37e+00	jre 1.6.0_24
	Node 6	Intel Xeon 3.2 G processor core x 2	6.32e+00	jre 1.6.0_24
Group 3	Node 7	AMD Opteron 2.8 processor core x 2	5.80e+00	jre 1.6.0_24
	Node 8	AMD Opteron 2.8 processor core x 2	5.82e+00	jre 1.6.0_24

Then the job client of Hadoop will submits the job (the JAVA program) from namenode and configure it to the jobtracker, then distributing the software to the slaves, scheduling the tasks and monitoring status of tasks, providing status and related

information to the job-client. Job-client will take the job, execute the assignment and feedback to namenode. Results of each job-client will send to namenode and shown in the logs in the sub-directory named "logs" of Hadoop directory.

All the nodes were connected to Nortel ERS 8600, it offers 512 Gigabits per second backplane switch capacity. Bandwidth connectivity between hosts was 1 gigabits each Ethernet port. Different hosts get various transmission rates from 765 Mbits/sec to 924 Mbits/sec, and the average transmission rate in this HDFS cluster is 851 Mbits/sec. It means that the networking performance of the HDFS cluster was under the optimal condition.

We use 1000 images, import them to the "input" directory that will be processed by the watermarking job. The size of each image is 512x512 pixels. The proposed watermarking algorithm, DDWT combined with SVD, will be executed over Hadoop architecture, in which the watermark embedding process and extracting process are executed for eight times, each time using different number of datanodes for the performance measurement.

4 Experimental Results

4.1 Calculation with Images Distributed Evenly

This section shows the experimental result. There were two methods to measure the execution time. The first method divides the 1000 images evenly, in this way, each node will process the same numbers of image, result of time consumed will wait for the slowest host to finish job. The results are listed in Table 2. Each datanode will have a different loading, depending on how many datanodes are used. For example, if five datanodes are used, then each datanodes will have 200 image files to be processed. Table 2 shows the number of segments that each datanode must process, along with the segment size for each slaves. Obviously, If all the image were processed in a single node, it spend 2,182 minutes to finished the watermarking job, whereas the execution time for the cluster with 8 nodes was 392 minutes.

Table 2. Executing results

Node number	Loading per node (Image numbers)	Time consumed (minutes)
1	**1000**	**2182**
2	**500**	**1003**
3	**333**	**718**
4	**250**	**523**
5	**200**	**592**
6	**167**	**525**
7	**143**	**445**
8	**125**	**392**

Figure 2 shows the executing results of method 1. There is another point we generated from Figure 2. Because the result of time consumed of each calculation

process will wait for the slowest host to return it's consumed time to program, when the slower one was added into the computing cluster, time consumed will increases and slope of the time-consumed line will becomes positive as Figure 2 since slower node 5 was added into computing cluster. The next experiment in section 4.2 shows how to resolve this issue.

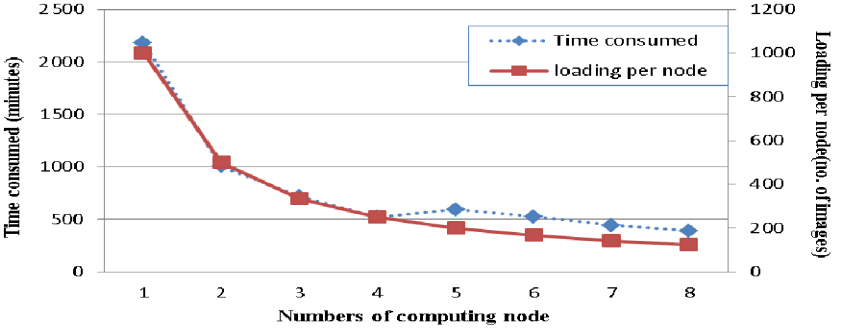

Fig. 2. Executing result chart

4.2 Calculation with Computing Ability Optimized

The second method distributes the 1000 images into different numbers to each host according to the HPC challenge benchmark, results will presents as gflops format (Please reference Table 1). Gflops or gigaflops represents a measurement in billions of floating-point operations per second (FLOPS) that a computer's microprocessor can handle. In this method that image numbers distributes by host's gflops, we optimized and sorting computing node from high to low for efficiency distribution.by their computing ability. The results are listed in Table 2. After the adjustment by gflops values generates from HPC challenge benchmark, node loading was optimized by their computing ability. Time consumed shown in Table 2 was better than Table 1 which distributed images evenly.

Obviously, in the first method, the computing group 1 (shown in Table 1, from node 1 to node 4) consumed the same period of time with second method to finish the watermarking job, whereas the best execution time for a cluster with 8 nodes is 392 minutes. From Table 2, distributed numbers of images were optimized by HPC challenge benchmark, the best execution time for a cluster with 5 nodes (shown in Table 3)will decrease from 592 minutes to 441 minutes, and the time consumed of the cluster computing with 8 nodes will decreases from 392 to 268, it means that image numbers distributed by the optimization of HPC challenge benchmark will enhances the efficiency of this implementation.

This implementation shows that when process these data with a single computer, it will consume quite a long period of time to complete this job. In the cloud environment, the impact will not be so much because all the jobs are distributed to multiple computing nodes.

From Figure 3, we deduce another interesting point, i.e., the curve of the execution time decreases slowly when the computing nodes are more than five. Due to the fact

that the execution time of eight computing nodes are very close to the point of convergence, it is assumed that eight computing nodes give the optimal solution for the watermarking algorithm combining SVD with DDWT.

Table 3. Executing results

computing node	Node number	Node Loading (Image numbers)	Time consumed (minutes)
Node 3	1	1000	2182
Node 3,4	2	500	1003
Node 3,4,7	3	333	718
Node 3,4,7,8	4	250	523
Node (3,4,7,8),5	5	(221x4),116	441
Node (3,4,7,8),(5,6)	6	(198x4),(112x2)	372
Node (3,4,7,8),(5,6),2	7	(181x4),(94x2),88	301
Node (3,4,7,8),(5,6),(1,2)	8	(167x4),(79x2),(87)	268

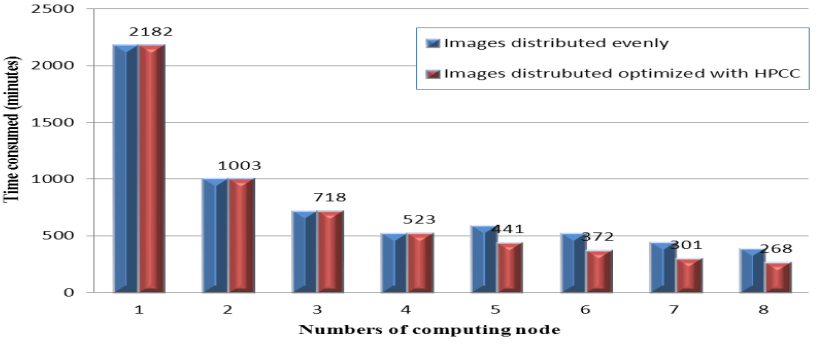

Fig. 3. Comparison chart

5 Conclusions and the Future Work

This implementation shows how to effectively process image watermarking based on a robust method which combines the Singular Value Decomposition and Distributed Discrete Wavelet Transformation over Hadoop computing environments. Through the implementation, we offer a better way to reduce the time consumed for such a heavy computing requirement.

Cloud environment is more and more popular and reliable. Issues of copyright infringement will accompany with the vast amount of personal files like pictures in Facebook or videos on YouTube. In the related social network website, the method we proposed can apply at all the digital files, such as pictures, videos, audios and also the documents in the cloud environment. Computation of digital watermarking technology over cloud environments will play a very important role for the protection of intellectual property in the age of digitalization.

Cloud environment is not only implemented for the public cloud, but also for lots of private cloud. Most of them are powered by Hadoop, which is the same cloud environment we used in this paper. In views of cost, performance and data security, business always chases on economies of scale for the balance of the lower cost, higher performance and more reliability. The proposed method and architecture will be helpful for enhancing data security in private companies, like online encryption for design drawing; and even more, be used to design a security gateway to detect if anyone trying to illegal upload the watermarked files to the Internet. So, the proposed method will play an important role in the future.

References

1. Andrews, H.C., Patterson, C.L.: Singular Value Decomposition (SVD) Image Coding. IEEE Transactions on Communications, 425–432 (1976)
2. Garguir, N.: Comparative Performance of SVD and Adaptive Cosine Transform in Coding Images. IEEE Transactions on Communications, 1230–1234 (1979)
3. Leary, D.P.O., Peleg, S.: Digital Image Compression by Outer Product Expansion. IEEE Transactions on Communications, 441–444 (1983)
4. Liu, R., Tan, T.: An SVD-based Watermarking Scheme for Protecting Rightful Ownership. IEEE Transactions on Multimedia, 121–128 (2002)
5. Tang, X., Yang, L., Yue, H., Yin, Z.: A Watermarking Algorithm Based on the SVD and Hadamard Transform. In: International Conference on Communications, Circuits and Systems, Hongkong, China, p. 877 (2005)
6. Ma, L., Li, C., Song, S.: Digital Watermarking of Spectral Images Using SVD in PCA-Transform Domain. In: IEEE International Symposium on Communications and Information Technology, Beijing, China, pp. 1489–1492 (2005)
7. Deng, T.B., Nakagawa, Y.: SVD-based Design and New Structures for Variable Fractional-delay Digital Filters. IEEE Transactions on Signal Processing, 2513–2527 (2004)
8. Redif, S., Cooper, T.: Paraunitary Filter Bank Design via a Polynomial Singular-Value Decomposition. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, Quebec, Canada, pp. 613–616 (2005)
9. Karkarala, R., Ogunbona, P.O.: Signal Analysis Using a Multiresolution Form of the Singular Value Decomposition. IEEE Transactions on Image Processing, 724–735 (2001)
10. Vozalis, M.G., Margaritis, K.G.: Applying SVD on Item-Based Filtering. In: 5th International Conference on Intelligent Systems Design and Applications, Wroclad, Poland, pp. 464–469 (2005)
11. Lin, C.H., Jen, J.S., Kuo, L.C.: Distributed Discrete Wavelet Transformation for Copyright Protection. In: 7th International Workshop on Image Analysis for Multimedia Interactive Services, Incheon, Korea, pp. 53–56 (2006)
12. The Apache™ Hadoop™ project, `http://hadoop.apache.org/`
13. Jaliya, E., Shrideep, P., Geoffrey, F.: MapReduce for Data Intensive Scientific Analyses. In: Proceedings of the IEEE Fourth International Conference on eScience, Indianapolis, USA, pp. 277–284 (2008)
14. Lin, C.H., Liu, J.C., Shih, C.H., Lee, Y.W.: A Robust Watermark Scheme for Copyright Protection. In: Multimedia and Ubiquitous Engineering, Busan, Korea, pp. 132–137 (2008)