Distributed Video Transcoding System based on MapReduce for Video Content Delivery

Myoungjin Kim', Hanku Lee[1]'z* Hyeokju Lee' and Seungho Han'

' Department of Internet and Multimedia Engineering, Konkuk University, Seoul, Korea
[2] Center for Social Media Cloud Computing, Konkuk University, Seoul, Korea
{tough105, hlee, hjlee09, shhan87}@konkuk.ac.kr

Abstract. In this paper, we propose a distributed video transcoding system that transcodes various video codec formats into the MPEG-4 video format. This system makes various kinds of video content available for heterogeneous devices such as the smart phones, PCs, TVs, and pads. We design and implement our system using the MapReduce framework running on a Hadoop Distributed File System (HDFS) platform and the media processing library Xuggler. In this way, we exponentially reduce the encoding time to transcode large amounts of video content, facilitating a transcoding function. Based on experiments performed on a 28-node cluster, the proposed distributed video transcoding system provides excellent performance in terms of speed and quality.

Keywords: Video Transcoding, MapReduce, Hadoop, and Distributed Processing

1 Introduction

Due to the explosive growth of mobile devices and improvements in mobile device communication technologies, media streaming services and applications based on video content have gained remarkable popularity and interest from users. Under these circumstances, transcoding techniques [1] are needed in order to deliver large amounts of video data in multiple formats to heterogeneous mobile devices.

The transcoding system requires a high performance computing capability as well as the intensive use of scalable computing resources. Video transcoding includes three main process steps: decoding, resizing, and encoding. However, the transcoding processing time is extremely slow since most existing transcoding systems run within a traditional single PC environment.

In order to reduce transcoding time significantly, we propose a distributed video transcoding system based on MapReduce [2] running on a Hadoop Distributed File System (HDFS) [2]. The proposed system is able to transcode a variety of video coding formats into the MPEG-4 video format. Improvements in quality and speed are realized by adopting a HDFS for storing large amounts of video data created from

* Corresponding author

numerous users, MapReduce for distributed and parallel processing of video data, and Xuggler [3] for transcoding based on an open source.

Our paper is organized as follows: Section 2 introduces Hadoop and MapReduce. In Section 3, we explain our distributed video transcoding system based on MapReduce with simple characteristics. Experiment and its results are provided in Section 4. Section 6 concludes our paper.

2 Hadoop and MapReduce

Hadoop, inspired by Google's MapReduce and Google File System, is a software framework that supports data-intensive distributed applications handling thousands of nodes and petabytes of data. It can perform scalable and timely analytical processing of large data sets to extract useful information. Hadoop consists of two important frameworks: 1) Hadoop Distributed File System (HDFS), like GFS, is a distributed, scalable and portable file system written in Java. 2) MapReduce is the first framework developed by Google for processing large data sets.

The MapReduce framework provides a specific programming model and a runtime system for processing and creating large data sets amenable to various real-world tasks. This framework also handles automatic scheduling, communication, and synchronization for processing huge datasets and it has fault tolerance capability. The MapReduce programming model is executed in two main steps called *mapping* and *reducing.* Mapping and reducing are defined by *mapper* and *reducer* functions. Each phase has a list of key and value pairs as input and output. In the *mapping* step, MapReduce receives input data sets and then feeds each data element to the mapper in the form of key and value pairs. In the *reducing* step, all the outputs from the *mapper* are processed, and the final result is generated by the *reducer* using the merging process. Figurel shows the example of word counting using MapReduce framework.

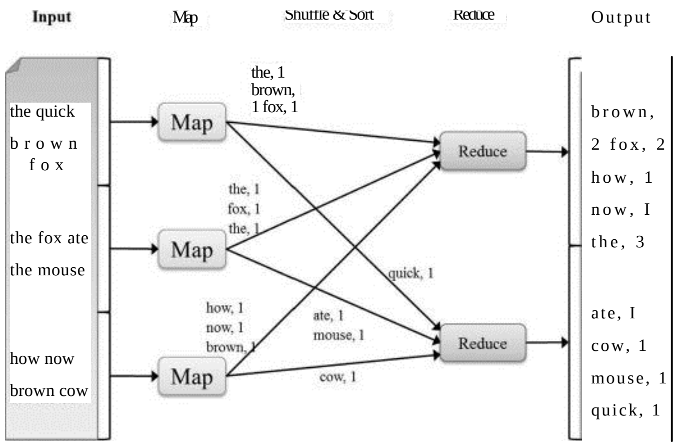
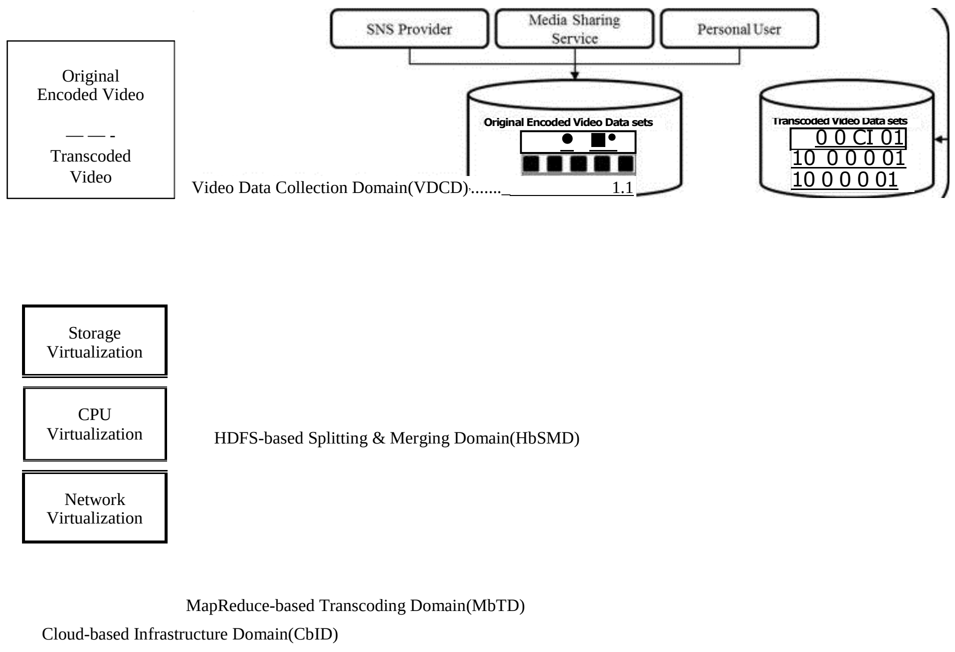


Fig. 1. An example of word counting using MapReduce

3 Distributed Video Transcoding System based on MapReduce

In this section, we briefly describe our proposed system architecture. 1) Our system contains a codec transcoding function and a function with a different display size, codec method, and container format. 2) Our system mainly focuses on the batch processing of large numbers of video files collected for a fixed period rather than the processing of small video files collected in real time. 3) HDFS is applied to our system in order to avoid the high cost of the communication of the video file while data transfer occurs for distributed processing. HDFS is also applied to our system due to the large chunk size (64MB) policy suitable for processing video files and the user-level distributed system. 4) Our system follows load balancing, fault tolerance, and merging and splitting policies provided from MapReduce for distributed processing.

The proposed system uses HDFS as storage for distributed parallel processing. The very large amount of collected data is automatically distributed in the data nodes of HDFS. For distributed parallel processing, the proposed system exploits the Hadoop MapReduce framework. In addition, Xuggler libraries for video resizing and encoding are utilized in Mapper. The Map function processes each chunked block of video data in a distributed and parallel manner. Figure 2 illustrates the overall system architecture of a distributed video transcoding system based on MapReduce.



SNS Provider
Media Sharing Service
Personal User
Original Encoded Video
Transcoded Video
Original Encoded Video Data sets
Transcoded Video Data sets
Video Data Collection Domain(VDCD)
1.1
Storage Virtualization
CPU Virtualization
Network Virtualization
HDFS-based Splitting & Merging Domain(HbSMD)
MapReduce-based Transcoding Domain(MbTD)
Cloud-based Infrastructure Domain(CbID)

Fig. 2. Overall system architecture of a distributed video transcoding system

4 Experiment and Results

Performance evaluation is conducted on a 28 node HDFS cluster consisting of 1 master node and 27 slave nodes (data node). Each node running on the Linux OS (CentOS 5.5) is equipped with two Intel Xeon 4 core 2.13GHz processors with 4GB registered ECC DDR memory and 1TB SATA-2. All nodes are interconnected with a 100Mbps Ethernet adapter. We also use Java 1.6.0_23, Hadoop-0.20.2, and Xuggler 3.4.

To verify the performance evaluation for encoding very large amounts of video files into target files, we prepare and use seven kinds of video data sets (Table 1) according to their volume sizes. We measure the total time to transcode each data set with the different MapReduce options of block size (default: 64MB) and block replications (default: 3EA). Table 2 lists the parameters for the original and target transcoded video file. Table 3 lists the measured transcoding times in seconds for different block size and replication.

According to Table 3, our system provides an excellent transcoding time for very large amounts of video files. For example, our system takes approximately 2000 sec (about 33 min) in order to complete the transcoding process in a 50G video data set with the default Hadoop options. In this experiment, we demonstrate convincingly that our system shows a better performance when block size is set to 256MB and 512MB or block replication is set to 3EA as compared to other conditions.

Table 1. Video data sets for performance evaluation

Parameter	1GB	2GB	4GB	8GB	10GB	20GB	50GB
Number of video files	5	10	20	40	50	100	250

Table 2. Parameters for each original and the transcoded video file

Parameter	Original video file	Transcoded video file
Codec	XviD	MPEG-4
Container	AVI	MP4
Size	200MB	60MB
Duration	3m19s	3m19s
Resolution	1280 x 720	320 x 240
Frame rate	29.97 fps	29.97 fps

Table 3. Total transcoding time according to the change of block size and the number of block replication (sec)

Block Size	1G	2G	4G	8G	10G	50G	Replication	1G	2G	4G	8G	10G	50G
32MB	182	269	366	617	741	3366	lEA	189	231	330	474	562	2661
64MB	171	183	232	378	438	1928	2EA	173	188	242	376	453	1944
128MB	109	124	177	219	275	983	3EA	171	183	232	378	438	1928
256MB	105	106	107	118	130	521	4EA	170	179	233	388	456	1966
512MB	107	109	110	119	129	515	5EA	229	232	287	395	468	2022

5 Conclusion

In this paper, we proposed a distributed video transcoding system based on MapReduce and HDFS that transcodes various video codec formats into the MPEG-4 video format. Our system ensures uniform transcoded video quality and a fast transcoding process. We experimentally verified the excellence of our system in video transcoding processing using distributed techniques.

Acknowledgments. This research was supported by the MKE (The Ministry of Knowledge Economy), Korea, under the ITRC (Information Technology Research Center) support program supervised by the NIPA (National IT Industry Promotion Agency (NIPA-2012 — (NIPA-2012-H0301-12-3006)).

References

1. Sandro Moiron, Mohammed Ghanbari, Pedro Assuncalo and Sergio Faria: Video Transcoding Technique, Studies in Computational Intelligence, vol. 231, pp. 245--270. Springer, Heidelberg (2009)
2. Wikipedia, http://en.wikipedia.org/wiki/Apache_Hadoop, 2001
3. http://www.xuggle.com/xuggler/