ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

클라우드 컴퓨팅을 이용한 유시티 비디오 빅데이터 분석☆

An Analysis of Big Video Data with Cloud Computing in Ubiquitous City

이 학 건[1] 윤 창 호[1] 박 종 원[1] 이 용 우[1*]
Hak Geon Lee Chang Ho Yun Jong Won Park Yong Woo Lee

요 약

유비쿼터스 시티(유시티)에서는 수많은 비디오 카메라들이 설치된다. 이렇게 설치된 많은 카메라로부터 대용량의 비디오 데이터가 실시간으로 끊임없이 발생하고 유시티의 관리 시스템으로 전달된다. 유시티의 다양한 서비스들을 뒷받침하기 위해서는 이러한 비디오 데이터를 저장하고, 이렇게 저장된 대용량의 비디오 데이터를 분석할 수 있는 방법과 관리 시스템이 요구된다. 그래서, 이 논문에서는 클라우드 컴퓨팅을 기반으로 한 유시티 비디오 관리 시스템을 제안한다. 또한, 근래 주목받고 있는 데이터 병렬처리 프레임워크인 Hadoop MapReduce를 이용하여 이러한 빅데이터 비디오를 분석하는 방법을 제안하고, 이에 따른 우리의 성능 평가를 소개한다.

☞ 주제어 : 유비쿼터스 시티, 유시티, 클라우드 컴퓨팅, 시스템 아키텍처, 맵리듀스, 하둡.

ABSTRACT

The Ubiquitous-City (U-City) is a smart or intelligent city to satisfy human beings' desire to enjoy IT services with any device, anytime, anywhere. It is a future city model based on Internet of everything or things (IoE or IoT). It includes a lot of video cameras which are networked together. The networked video cameras support a lot of U-City services as one of the main input data together with sensors. They generate huge amount of video information, real big data for the U-City all the time. It is usually required that the U-City manipulates the big data in real-time. And it is not easy at all. Also, many times, it is required that the accumulated video data are analyzed to detect an event or find a figure among them. It requires a lot of computational power and usually takes a lot of time. Currently we can find researches which try to reduce the processing time of the big video data. Cloud computing can be a good solution to address this matter. There are many cloud computing methodologies which can be used to address the matter. MapReduce is an interesting and attractive methodology for it. It has many advantages and is getting popularity in many areas. Video cameras evolve day by day so that the resolution improves sharply. It leads to the exponential growth of the produced data by the networked video cameras. We are coping with real big data when we have to deal with video image data which are produced by the good quality video cameras. A video surveillance system was not useful until we find the cloud computing. But it is now being widely spread in U-Cities since we find some useful methodologies. Video data are unstructured data thus it is not easy to find a good research result of analyzing the data with MapReduce. This paper presents an analyzing system for the video surveillance system, which is a cloud-computing based video data management system. It is easy to deploy, flexible and reliable. It consists of the video manager, the video monitors, the storage for the video images, the storage client and streaming IN component. The "video monitor" for the video images consists of "video translater" and "protocol manager". The "storage" contains MapReduce analyzer. All components were designed according to the functional requirement of video surveillance system. The "streaming IN" component receives the video data from the networked video cameras and delivers them to the "storage client". It also manages the bottleneck of the network to smooth the data stream. The "storage client" receives the video data from the "streaming IN" component and stores them to the storage. It also helps other components to access the storage. The "video monitor" component transfers the video data by smoothly streaming and manages the protocol. The "video translator" sub-component enables users to manage the resolution, the codec and the frame rate of the video image. The "protocol" sub-component manages the Real Time Streaming Protocol (RTSP) and Real Time Messaging Protocol (RTMP). We use Hadoop Distributed File System(HDFS) for the storage of cloud computing. Hadoop stores the data in HDFS and provides the platform that can process data with simple MapReduce programming model. We suggest our own methodology to analyze the video images using MapReduce in this paper. That is, the workflow of video analysis is presented and detailed explanation is given in this paper. The performance evaluation was experiment and we found that our proposed system worked well. The performance evaluation results are presented in this paper with analysis. With our cluster system, we used compressed 1920x1080(FHD) resolution video data, H.264 codec and HDFS as video storage. We measured the processing time according to the number of frame per mapper. Tracing the optimal splitting size of input data and the processing time according to the number of node, we found the linearity of the system performance.

☞ keyword : Ubiquitous City, U-City, Cloud Computing, System Architectures, MapReduce, Hadoop.

[1] School of Computer Science and Engineering, University of Seoul, Seoul, 130-743, Korea.
* Corresponding author (ywlee@uos.ac.kr)
[Received 31 July 2013, Reviewed 19 August 2013, Accepted 11 April 2014]

☆ 본 논문은 2013년도 인터넷정보학회 추계학술발표대회 우수논문 추천에 따라 확장 및 수정된 논문임.

1. 서 론

유비쿼터스 시티(유시티)는 첨단 정보통신 인프라를 통하여 교량, 도로, 건물 등의 도시 인프라가 유비쿼터스 정보서비스를 통해 융합하여 만들어진 미래의 첨단 도시이다. 유시티는 도시 생활의 편의 증대와 삶의 질 향상, 효율적인 도시관리 등을 제공하는 것을 목표로 한다. 이런 유시티 내에서 시민들은 다양한 서비스(환경, 교통, 재난 정보 등)을 이용하며 한층 나아진 생활을 경험할 수 있다[1].

이런 유시티 내부에서는 다양한 유시티 서비스를 지원하기 위하여 수많은 비디오 카메라와 센서들이 설치되어 있다. 도시단위의 이러한 카메라와 센서에서 발생되는 데이터의 양은 굉장히 막대하며, 이러한 많은 양의 카메라와 센서를 관리해야한다[2]. 유시티는 중앙 제어 센터를 통해서 이러한 인프라들을 관리하고 데이터들을 처리해야 하는데 일반적인 방법으로는 이러한 문제들을 다루기 힘들다. 이러한 요구사항들은 클라우드 컴퓨팅이 보여주는 장점들을 통하여 극복이 가능하다[3][4].

클라우드 컴퓨팅은 현재 가장 주목받고 있는 기술이다. Vmware와 Xen, Eucalyptus와 같은 운영체제 레벨 가상화 기술은 리소스를 사용하는 새로운 방법을 제시하여 주었다. 이러한 방법은 Amazon, Google 등과 같은 IT기업들의 클라우드 데이터센터를 등장시켰고, 이런 데이터 센터들로부터 일반 유저들도 쉽게 리소스를 공급받을 수 있게 되었다. 또, 하드웨어를 원하는 만큼의 리소스로 나눌 수 있기 때문에 리소스의 관리 및 배치 등을 자유롭게 해주었다.

이렇게 일반 유저들이 대용량의 컴퓨팅 리소스를 원하는 만큼 빌려쓸 수 있게 됨에 따라 확장성 있는 시스템과 솔루션에 대한 요구들이 등장하였다. 이러한 무한한 확장성을 토대로 클라우드 컴퓨팅을 기반으로 한 서비스들을 쉽게 구축할 수 있도록 Hadoop[4], Cassandra, MongoDB와 같은 다양한 오픈소스들이 개발되고 있다.

그 중에서 Hadoop[5]은 가장 주목 받고 있는 오픈소스이다. Hadoop은 대용량 데이터 처리를 위해서 개발된 오픈소스로, HDFS(Hadoop Distributed File system)에 대용량의 데이터를 일정한 청크 단위로 분할하여 저장하고 복제본(Replication)을 통하여 데이터의 백업 및 읽기 성능을 향상시켰다. 이렇게 분할되어 저장된 데이터를 MapReduce[6]를 이용하여 단순화된 알고리즘을 통하여 분석할 수 있는 플랫폼을 제공해주었고, 장애 극복(Fault-tolerance)를 지원해 주기 때문에 프로그래머는 분석 알고리즘에 집중할 수 있도록 해주었다.

이를 통하여 전까지는 많은 양으로 인해 분석하기 어려웠던 비정형 데이터를 분석할 수 있게 해 주었다. 비디오는 이러한 비정형 데이터의 대표적인 예 중 하나지만, 아직까지 MapReduce 등을 이용하여 분석하려는 시도는 굉장히 드물었다.

그래서 본 논문에서는 빅데이터 비디오에 대한 문제들을 해결하기 위한 클라우드 컴퓨팅 기반의 확장성 있는 시스템 아키텍처를 제시하고, 저장된 비디오를 MapReduce를 통해 분석하고, 그에 따른 성능평가를 통해 유용성을 증명하고자 한다.

본 논문의 구성은 2장에서는 관련 연구를 살펴보고, 3장에서는 우리가 설계한 시스템 아키텍쳐를 서술한다. 4장에서는 MapReduce를 통한 비디오 분석 기법에 대하여 설명한다. 5장에서는 실험과 성능평가를 6장에서는 결론 및 향후 연구에 대하여 논한다.

2. 관련 연구

2.1 비디오 시스템과 클라우드 컴퓨팅

도시 내에서 비디오를 분석하고 이를 활용하는 가장 널리 쓰이는 대표적인 시스템은 비디오 감시 시스템을 예로 들 수 있다. 테러 감지, 재난 감지, 건물 관리, 환경 모니터링 등의 다양한 용도로 쓰일 수 있다. 유시티 내에서도 효율적인 관리와 다양한 유시티 서비스들을 위해서는 이러한 비디오 시스템은 필수적으로 요구되며 충족되어야할 사항이다.

하지만 현재 이러한 비디오 감시 시스템 및 비디오 플랫폼들은 지속적으로 발생하는 많은 양의 비디오로 인해 다양한 문제와 맞닿아 있다. 비디오의 고화질화와 많은 양으로 인한 스토리지 확장의 문제, 관리 및 집중을 위한 중앙센터 구축의 문제, 대량의 데이터를 처리하는 프로세싱 파워, 데이터 마이닝과 정보통합 등의 다양한 문제에 맞서 과거에는 분산시스템을 통하여 이런 문제들에 대한 연구가 이루어졌다[7][8][9]. 이제 도시단위에 가까운 시스템을 위하여 클라우드 컴퓨팅 도입을 통해 이러한 문제들을 해결하려는 연구들이 진행되고 있다.

클라우드 컴퓨팅의 확장성을 이용하여 네트워크 비디오 레코더(NVR, Network Video Recorder)와 디지털 비디오 레코더(DVR, Digital Video Recorder)과 같은 기존의 장비들을 클라우드 기반의 레코더로 대체하는 연구[10], 기

존의 비디오 감시 아키텍쳐를 클라우드 컴퓨팅 기반으로 웹 서버, 클라우드 프로세싱 서버, 클라우드 스토리지 서버의 3부분으로 구성하여 설계한 연구[11], 집중되는 비디오 데이터로 인해 일어나는 네트워크 병목현상을 해결하기 위해 클라우드 컴퓨팅과 P2P(peer to peer)와 HDFS를 이용한 연구[12], 클라우드 기반의 비디오 플랫폼을 위한 가상 머신(VM)에 따른 효율적인 배치를 위한 연구[13], 비디오 시스템의 클라우드 컴퓨팅 도입에 따른 신뢰성 검증 연구[14] 등 다양한 연구들이 이루어지고 있다.

기존의 비디오 시스템들은 분산시스템을 이용하여 각각의 분산된 장소에서 비디오 데이터를 처리하였다. 그리고 처리된 이벤트 데이터만을 중앙에서 처리하는 방식을 택하였다[7][8][9]. 하지만, 위와 같이 비디오를 클라우드 스토리지에 저장하는 연구, 이를 위한 네트워크와 가상머신의 효율적인 활용 등도 점차 연구되고 있다. 이에 따라 중앙집중화되어 저장된 클라우드 컴퓨팅을 이용한 비디오 데이터의 처리에 대한 연구가 필요하다.

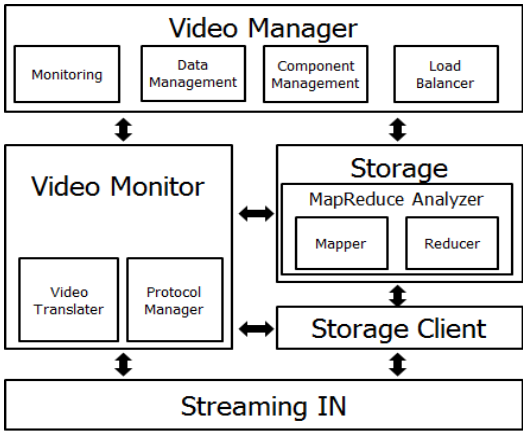
2.2 MapReduce를 이용한 비디오 빅데이터 처리 연구

MapReduce 프로그래밍 모델은 Map과 Reduce 2단계로 이루어진 간단한 알고리즘으로, Google에서 자신들의 데이터 분석에 MapReduce 알고리즘을 사용하는 것을 발표되고, Hadoop과 같은 오픈소스 구현체들이 등장함에 따라 주목받고 점차 다양한 목적으로 확대되었다.

처음에는 MapReduce는 로그분석과 같은 text기반의 작업을 처리하는 위주로 사용되었지만, MapReduce를 이용하는 분야가 점점 증가함에 따라 비디오 처리에 대한 연구도 조금씩 진행되고 있다. [15] 에서는 무압축 비디오를 H.264 코덱을 이용하여 압축하는데 사용하는 아키텍쳐가 제시하였다. [16] 에서는 MapReduce에 기반한 다양한 컴퓨터 비전 알고리즘을 제안하였다. 하지만 이와 같은 연구는 무압축 영상이나 여러 장의 이미지만을 사용하여 실질적으로 사용되는 인코딩된 영상에 대한 고려가 부족하다고 볼 수 있다.

3. 시스템 아키텍쳐 : 유시티 비디오 시스템

우리가 디자인한 유시티 비디오 시스템 아키텍처는 그림 1과 같다. 이 시스템은 클라우드 컴퓨팅 내에서 손쉬운 배포와 유연하고 신뢰성을 확보하는 것을 목적으로 설계되었다. 모든 컴포넌트는 비디오 시스템 내에서의 각 기능적 요구사항에 맞춰 설계되었다.



(그림 1) 유시티 비디오 시스템 아키텍처
(Figure1) Architecture of the U-City Video System

또한, 이 시스템은 들어온 비디오 카메라 데이터를 받아 관리하고 처리하며 [19], 유시티 포탈의 비디오 매니저를 통하여 중앙관리 센터나 다중의 유저, 다양한 유저 디바이스로 표시해준다.

- 스트리밍 인(Streaming IN) 컴포넌트는 네트워크 카메라와 센터로부터 오는 데이터를 받는 역할을 수행한다. 또한, 인터넷과 같은 공용망에서 인트라 넷과 같은 사설망으로 비디오 데이터를 전달하는 역할도 수행한다. 대용량의 스트림을 처리하기 위하여 네트워크의 병목현상을 관리하는 기능도 수행한다.
- 스토리지 클라이언트(Storage Client)는 클라우드 스토리지의 클라이언트로 스트리밍 인 컴포넌트로부터 받은 비디오 스트림을 저장한다. 또한, 다른 컴포넌트들이 클라우드 스토리지에 접근하는 것을 도와주는 역할을 수행한다.
- 비디오 모니터(Video Monitor) 컴포넌트는 선택된 비디오를 클라우드 스토리지로부터 읽어오거나 스트리밍 인 컴포넌트로부터 받아 스트리밍을 해준다. 내부 컴포넌트로 비디오 트랜스레이터(Video Translator)와 프로토콜 매니저(Protocl Manger)를 가지고 있다. 첫째로, 비디오 트랜스레이터는 해상도와 코덱, 프레임레이트를 사용자의 요청에 따라 바꿔주는 역할을 담당한다. 둘째로, 프로토콜 매니저는 RTSP(Real Time Streaming Protocol)와 RTMP(Real Time Messaging Protocol)와 같은 비디오 스트리밍 프로토콜들을 관리해주는 역할을 담당한다.

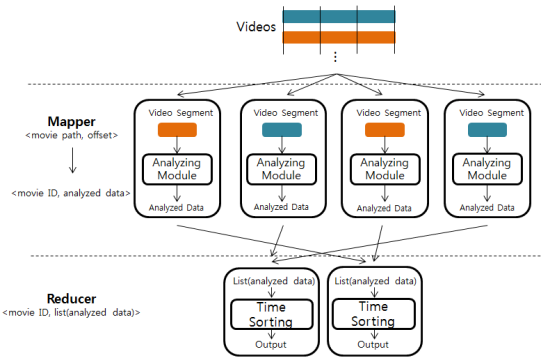
- 비디오 매니저(Video Manager)는 다른 컴포넌트들을 관리하고 데이터 관리 및 로드 밸런싱, 포탈을 통해 비디오 모니터링 기능을 제공한다.

이 연구에서는 클라우드 스토리지로 HDFS를 이용하였다. 다음 장에서는 클라우드 스토리지에 저장된 대용량 비디오 데이터를 처리하기 위해 구현한 MapReduce Analyzer와 비디오 분석 방법에 대하여 서술한다.

4. 워크 플로우

이 시스템에서 구현한 MapReduce Analyzer의 과정은 그림 2와 같다. 먼저 일정한 시간간격으로 비디오를 나눠 Map의 입력으로는 비디오 경로와 자른 시간의 오프셋 값을 키와 밸류로 설정하였다. 이렇게 넘겨진 비디오 경로와 시간 오프셋 값은 각각의 Mapper로 전달된다. Mapper에서는 해당 비디오의 경로와 시간 오프셋 값을 이용해 정해진 길이만큼 비디오를 잘라온다. 그 후, 잘라온 비디오를 분석 모듈(Analyzing Module)을 이용하여 처리한다. 그 후, 입력으로 받은 비디오 경로에서 잘라온 비디오 이름을 이용하여 <비디오 ID, 분석 데이터>의 쌍으로 Mapper의 출력 값으로 출력된다.

Mapper에서 출력된 중간 데이터들은 키로 설정된 각각의 비디오 ID별로 모여서 정렬되고, Reduce는 <비디오 ID, list(분석 데이터)>의 형태로 입력으로 받게 된다. 하지만 이렇게 넘겨받은 데이터는 아직 시간 순으로 정렬되지 않은 상태이다. 그래서 Reducer는 분석 데이터들을 시간 축에 맞게 재배열하여 출력한다. 표1은 MapReduce에서 앞에서 설명한 <키, 밸류> 값을 정리한 것이다.



(그림 2) MapReduce 비디오 분석 흐름도
(Figure2) Workflow of Video Analysis using MapReduce

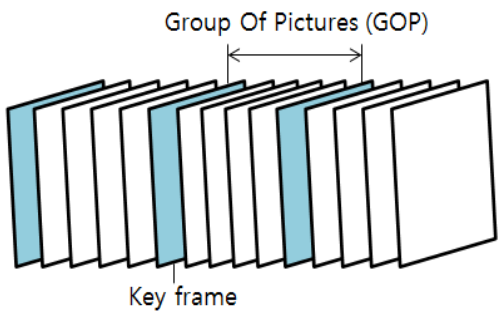
이러한 MapReduce 과정에서 비디오를 어떻게 자를 것인가는 굉장히 큰 이슈이다. [15]에서는 무압축 영상을 HDFS의 저장된 블록 단위로 잘라 처리하였다. 그러나 이 연구에서는 코덱을 이용한 압축된 영상을 사용할 것이기 때문에 압축된 영상의 특성을 고려하여야 한다.

(표 1) MapReduce 〈키, 밸류〉 설정

(Table1) The Definition of Key-Value pair in MapReduce

Map 단계	입력	<비디오 경로, 시간 오프셋 값>
	출력	<비디오 ID, 분석 데이터>
Reduce 단계	입력	<비디오 ID, list(분석 데이터)>

현재 가장 많이 쓰이는 코덱인 H.264와 같은 MPEG계열의 코덱으로 영상을 압축할 때는 GOP(Group Of Pictures) 단위로 영상이 압축된다. 그림 3과 같이 GOP는 압축된 이미지(프레임)들의 모임이다. 이런 프레임들의 모임은 Key 레임과 Non-Key 프레임으로 구성되어 있다. 다른 프레임을 참조하지 않고 독립적으로 복원되는 프레임을 I-frame(Intra Frame) 혹은 Key 프레임이라고 한다.



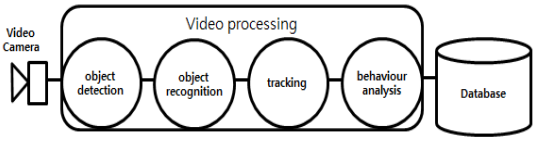
(그림 3) Group Of Pictures (GOP)
(Figure3) Group of Pictures (GOP)

비디오를 자를 때 GOP를 고려하지 않고 자를 경우 Key 프레임이 유실되면서 해당 Key 프레임을 참조하는 프레임들을 읽을 수 없게 되고 비디오 데이터가 손상된다. 이를 피하기 위해서는 손상되는 부분을 다시 인코딩하는 방법도 있지만 인코딩 작업은 처리시간이 많이 걸리는 작업으로 비효율적이 된다. 그래서 이 연구에서는 키 프레임이 유실되는 것을 막기 위해 일정한 GOP 길이에 따라 비디오를 잘라오도록 하였다.

비디오의 헤더 또한 고려해야 되는 문제이다. 중간의 비디오를 잘라 그대로 사용할 경우, 헤더가 없는 영상이 된다. 그로 인해 헤더에 저장된 영상의 다양한 정보가 유실되어 비디오를 읽을 수 없는 문제가 발생하거나, 비디오를 읽어오기 위해 헤더의 정보를 분석모듈 마다 전달하여야 하는 과정이 추가되어야한다. [15]에서는 이러한 문제를 해결하기 위해 헤더를 포함한 컨테이너를 미리 준비하여 그 안에 잘라온 비디오를 담았다. 여기서도 비슷한 방법을 이용하여 비디오를 잘라오면서 같은 정보의 헤더를 다시 붙이는 방법을 사용하였다. 이로 인해 분석모듈은 독립적으로 작동이 가능하여 교체가 가능하고, 유연한 개발이 가능하도록 하였다.

5. 성능평가

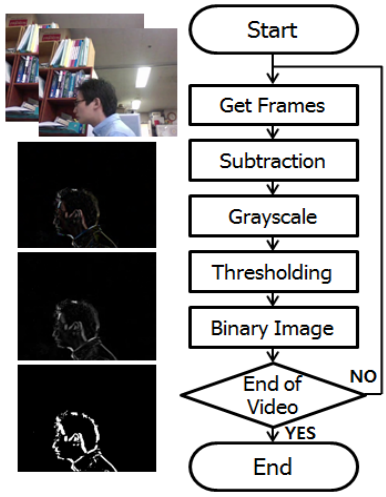
컴퓨터 비전의 영역에서는 정말 다양한 알고리즘이 존재한다. 이 수많은 알고리즘 중에서 가장 기본이 되는 알고리즘을 성능평가를 위해 택하였다. 그림 4는 일반적인 비디오 감시 시스템의 흐름도를 나타낸 것이다. 비디오 처리의 가장 첫 단계로 객체 인식(object detection)을 볼 수 있다. 객체 인식을 통하여 물체를 감지하고 그 후 그 물체를 인식하고, 추적, 행동 분석을 하여 의미있는 정보를 추출한다고 볼 수 있다 [18].



(그림 4) 일반적인 비디오 감시 시스템의 흐름도 [9]
(Figure4) Traditional Flow of Processing in Video Surveillance System [9]

객체 인식에서는 시차(temporal difference) 영상을 이용하는 방법과 배경 제거 기법(background subtraction)의 2가지 대표적인 방법이 있다. 2가지 모두 차영상(difference image)를 이용한 방법으로 2개의 다른 이미지의 차연산을 통한 방법이다. 시차 영상의 경우는 연속된 이미지를 사용하고, 배경 제거 기법의 경우는 객체가 없는 배경 이미지와 현재 이미지를 이용한다. 배경 제거 기법의 경우 계산 시간이 시차 영상을 구하는 것보다 빠르지만 바람에 흔들리는 나무 등과 같은 움직임이 많은 환경에서는 제대로 작동하지 못한다[9][19]. 이 성능평가에서는 계산량이 좀 더 많이 필요한 시차 영상을 이용하였다.

그림 5는 성능평가에서 사용한 시차 영상의 흐름도이다. 연속되는 프레임의 차영상을 구한 후, 이를 그레이스케일(grayscale)로 변환하고 스레쉬홀드(threshold) 값을 통하여 이진 영상(binary image)으로 변환시켜 객체를 추출하는 알고리즘을 구현하여 사용하였다.



(그림 5) 시차 영상(Temporal difference)의 흐름도
(Figure5) Workflow of Temporal difference

이 시스템에서 사용한 소프트웨어 세팅은 다음과 같다. 어플리케이션은 아파치 하둡 0.23.0 버전으로 Java 언어로 작성되었다. H.264 코덱으로는 오픈소스인 x264가 사용되었고, 비디오를 다루는 라이브러리로 FFmpeg의 libavutil과 ffprobe 라이브러리가 사용되었다. 분석 모듈은 가장 널리 쓰이고 있는 컴퓨터 비전 라이브러리인 OpenCV 2.4.3 버전으로 C++언어로 작성되었다. 각 라이브러리들의 의존성 문제로 인하여 모든 라이브러리들은 모든 노드에 설치되었다.

실험 환경으로는 14대의 PC 클러스터가 사용하였다. PC는 CPU로 i5 760 2.8Ghz 쿼드 코어 프로세서, 메인 메모리는 DDR3 8GB 램을 사용하였고, 하드디스크는 7200rpm의 500GB를 사용하였다. 모든 노드는 1기가비트 이더넷 스위치를 통하여 네트워크에 연결되어 있다. 운영체제는 Ubuntu 12.04 LTS 64bit server edition을 이용하였고, 자바 가상머신은 JVM 1.6.0_31 버전을 이용하였다. 모든 노드는 hdfs-fuse를 통해 HDFS가 마운트되어 있다.

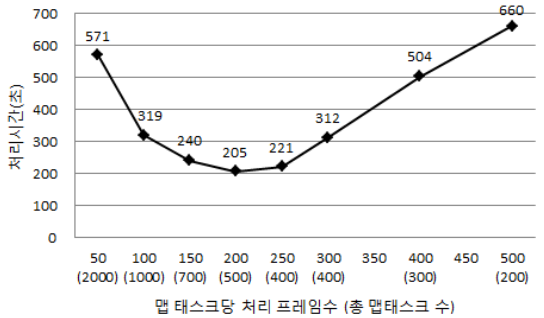
각 노드의 Mapper 개수는 3개, Reducer 개수는 4개로 설정하였다. Mapper를 4개로 설정할 경우 hdfs-fuse가 제대로 작동하지 못하는 문제로 인해 3개로 설정하였다.

입력 데이터로는 1920x1080 해상도(FHD)의 영상을 이용하였다. 총 길이는 1000 프레임으로, 비트레이트는 19.4Mbit/s, GOP의 길이는 25 프레임으로 설정하였다. 비디오 파일의 크기는 약 41MB로 이러한 비디오 파일을 100개를 사용하여 총 100,000 프레임을 처리하도록 하였다. 표2는 MapReduce 없이 1 대의 노드에서 프레임 수에 따른 처리 시간을 나타낸 것이다.

(표 2) 프레임 수에 따른 1대의 노드에서 처리 시간

(Table2) Processing Time in Single Node According to the Number of Frame

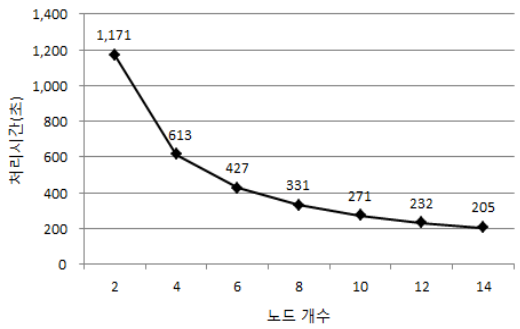
프레임 수	2	200	1000	100,000
처리 시간(초)	0.03555	7.0645	35.51945	3554.964



(그림 6) 맵 태스크 당 프레임 수에 따른 처리시간

(Figure6) Processing Time According to the Number of Frame per Mapper

그림 6은 한 개의 맵 태스크에 할당하는 프레임 개수를 변화시켜 가며 실험한 그래프이다. 처리시간은 실험을 총3번을 하여 평균값을 내어 측정하였다. 1개의 맵태스크당 할당하는 프레임 수가 작으면 작을수록 입력은 잘게 쪼개져서 태스크의 개수가 늘어나게 된다. 프레임 수를 작게 하여 잘게 쪼갤수록 시간이 줄어드는 듯 보이지만 200 이하로 잘라서 처리할 경우 시간이 오히려 늘어나는 것을 확인 할 수 있다. 너무 잘게 자를 경우, 맵 태스크의 개수가 너무 많아져 오히려 영상을 처리하는 시간보다 MapReduce를 작동하는데 더 많은 시간이 든다는 것을 확인 할 수 있다. MapReduce를 이용하여 작업을 할 경우 맵에 적절한 처리시간을 가진 양을 하나의 태스크로 할당해야 됨을 알 수 있다. 또한, 아주 작은 처리량의 맵 태스크를 여러 개 처리할 경우 하둡이 적절하지 않음을 보여주고 있다.



(그림 7) 노드 개수에 따른 처리 시간의 변화

(Figure7) Processing Time According to the Number of Node

그림 7은 실험을 통해 얻은 가장 빠른 처리시간을 보이는 맵 태스크 당 200 프레임을 처리하게 하여 노드 개수를 2대씩 변화시키며 실험한 결과이다. 처리시간은 실험을 총3번을 하여 평균값을 내어 측정하였다. 노드가 추가됨에 따라 처리시간이 반비례하게 줄어듬을 확인할 수 있었다. 1대의 노드에서 100,000 프레임의 처리시간을 3554초(59분 14초)에서 노드 14대를 이용하여 205초(3분 25초)까지 단축할 수 있었다.

6. 결론 및 향후 연구 방향

본 논문에서는 유시티를 위한 클라우드 컴퓨팅 기반의 비디오 데이터 관리 시스템을 디자인하였다. 또한, MapReduce를 이용하여 비디오를 처리하는 방법을 제시하고 그에 따른 성능 평가를 제시하였다. 클라우드 컴퓨팅을 이용하여 이러한 확장성 있는 시스템과 대용량 비디오 처리방법은 도시규모의 서비스를 구축할 때 매우 유용하게 쓰일 것으로 기대된다.

본 논문의 성능평가에서는 1920x1080의 FHD 영상만을 사용하였지만 앞으로 다양한 해상도의 비디오 및 해상도 외의 다양한 요소들을 고려하여 실험해 볼 예정이다. 그리고 HDFS를 스토리지로 사용하였지만, 비디오의 입력이 늘어날수록 스토리지의 성능이 중요한 요소로 작용하기 때문에 GlusterFS, Lustre, Ceph과 같은 다양한 분산 스토리지 솔루션을 이용한 실험을 통해 성능을 비교하여 최적의 솔루션을 찾도록 할 것이다. 또한, 제시한 MapReduce 방법의 최적화를 통하여 더욱 성능을 끌어올리기 위한 방법을 모색함과 동시에 Hadoop 이외의 CGL-MapReduce, Spark와 같은 병렬프레임워크에서도 성능을 평가할 것이다.

추후, 유시티의 다양한 시나리오에 맞는 비디오 알고리즘의 개발을 통하여 대용량 비디오의 분석을 통하여 의미있는 정보를 추출하고 이를 유시티 미들웨어 의 상황인식 컴퓨팅(Context-aware computing) 기술을 통해 추론하는 새로운 기술을 개발할 것이다. 이런 새로운 기술을 이용하여 차세대의 새로운 유시티 어플리케이션과 서비스를 개발할 수 있을 것으로 기대된다.

Acknowledgement

이 논문은 2011년도 서울시립대학교 교내학술연구비에 의하여 지원되었음.

참 고 문 헌(Reference)

[1] Cho, B.S., Jeong, W.S., Cho, H.S., “A Study on the Business and Trend of u-City”, Electronics and Telecommunications Trends,, vol. 21, no. 4, pp. 152-162, 2006.

[2] Burn, U.I., Nam, Y.Y., Cho, W.D., ″Agent-based Automatic Camera Placement for Video Surveillance Systems″ Journal of Korean Society for Internet Information, vol. 11, no.1, pp.103-116, 2010.

[3] Jung, H.S., Jeong, C.S., Lee, Y.W. and Hong P.D., ″An Intelligent Ubiquitous Middleware for U-City: SmartUM,″ Journal of information science and engineering, vol. 25, pp.375-388, 2009.

[4] Jung, H.S., Baek, J.K., Jeong, C.S., Lee, Y.W. and Hong, P.D., “Unified Ubiquitous Middleware for U-City,” Proceedings of the International Conference on Convergence Information Technology 2007 (ICCIT 07), pp.2347-2379, 2007.

[5] Apache Hadoop [Online] Available: http://hadoop.apache.org/

[6] Dean, J. and Ghemawat, S., ″MapReduce: Simplified Data Processing on Large Clusters,″ Proceedings of OSDI, pp.137-150, 2004.

[7] Detmold, H., Hengel, A.V.D., Dick, A.R., Falkner, K.E., Munro, D.S. and Morrison, R., ″Middleware for Distributed Video Surveillance,″ IEEE Distributed systems Online, vol.9, no.2, pp. 1-10, 2008.

[8] Hampaur, A., Borger, S., Brown, L., Carlson, C., Connell, J., Lu, M., Senior, A., Reddy, V., Shu, C. and Tian, Y., “S3 : The IBM Smart Surveillance System: From Transactional Systems to Observational Systems,” Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing 2007 (ICASSP 07), vol. 4, pp.1385-1388, 2007.

[9] Valera, M. and Velastin, S.A., “Intelligent distributed surveillance systems: a review,” Proceedings of the IEE Vision, Image and Signal Processing, vol. 152, no. 2, pp.192-204. 2005.

[10] Lin, C.F. Yuan, S., Leu, M. and Tsai, C., ″A Framework for Scalable Cloud Video Recorder System in Surveillance Environment,“ Proceedings of the Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC 12), pp.655-660, 2012.

[11] Rodriguez-Silva, D.A., Adkinson-Orellana, L., Gonz’lez-Castaño, F.J., Armino-Franco, I. and Gonz’lez-Martinez, D., ″Video Surveillance Based on Cloud Storage,″ Proceedings of the IEEE Cloud Computing (CLOUD 12), pp.991-992, 2012.

[12] Wu, Y.S., Chang, Y.S., Jang, T.Y. and Yen, J.S., “An Architecure for Video Surveillance Service based on P2P and Cloud Computing,” Proceedings of the Ubiquitous Intelligence & Computing and 9th International Conference on Autonomic & Trusted Computing (UIC/ATC 12), Sept. 2012, pp.661-666.

[13] Hossain, M.S., Hassan, M.M., Qurishi, M.A. and Alghamdi, A., ″Resource Allocation for Service Composition in Cloud-based Video Surveillance Platform,″ Proceedings of the IEEE International Conference on Multimedia and Expo Workshops (ICMEW 12), pp.408-412, 2012.

[14] Karimaa, Aleksandra, ″Video Surveillance in the Cloud: Dependability Analysis,″ Proceeding of the 4th International Conference on Dependability (DEPEND 11), pp. 92-95, 2011.

[15] Pereira, R., Azambuja, M., Breitman, K. and Endler, M., ″An Architecture for Distributed High Performance Video Processing in the Cloud,″ Proceedings of the IEEE 3rd International Conference on Cloud Computing (CLOUD 10), pp.482-489, 2010

[16] White, B., Yeh, T., Lin, J. and Davis, L., “Web-Scale Computer Vision using MapReduce for Multimedia Data Mining,” Proceedings of the 10th International Workshop on Multimedia Data Mining (MDMKDD 10), pp.1-10, 2010.

[17] Kim, Y.B., Kim. T.H., Lee, D.G., Kim. J.J., "Performance improvement for Streaming of High Capacity Panoramic Video" Journal of Korean Society for Internet Information, vol. 11, no.2, pp.143-153, 2010.

[18] Nam, Y.Y., Choi, Y.J, Cho, W.D, "Human Activity Recognition using an Image Sensor and a 3-axis Accelerometer Sensor" Journal of Korean Society for Internet Information, vol. 11, no.1, pp.129-141, 2010.

[19] Kim, M.Y., Jeon, H.S., Yeom, J.Y., Park, H.J., "An Improvement for Location Accuracy Algorithm of Moving Indoor Objects" Journal of Korean Society for Internet Information, vol. 11, no.2, pp.61-72, 2010.

◐ 저 자 소 개 ◑

이 학 건 (Hak Geon Lee)
2011년 서울시립대학교 전자전기컴퓨터공학부 (공학사)
2011~현재 서울시립대학교 전자전기컴퓨터공학과 석사과정
관심분야 : 클라우드 컴퓨팅, 병렬처리, 분산시스템, 시스템 프로그래밍, etc.
E-mail : withteas@uos.ac.kr

윤 창 호 (Chang Ho Yun)
2008년 서울시립대학교 전자전기컴퓨터공학부 (공학사)
2010년 서울시립대학교 전자전기컴퓨터공학과 (공학석사)
2010년~현재 서울시립대학교 전자전기컴퓨터공학과 박사과정
관심분야 : 클라우드 컴퓨팅, 유시티, 시멘틱웹
E-mail : touch011@uos.ac.kr

박 종 원 (Jong Won Park)
2009년 서울시립대학교 전자전기컴퓨터공학부 (공학사)
2011년 서울시립대학교 전자전기컴퓨터공학과 (공학석사)
2011년~현재 서울시립대학교 전자전기컴퓨터공학과 박사과정
관심분야 : 분산시스템, 클라우드 컴퓨팅, 시스템 프로그래밍
E-mail : comics77@uos.ac.kr

이 용 우 (Yong Woo Lee)
1981년 서울대학교 전기&컴퓨터 (공학사)
1990년 영국 Univ of Edinburg (공학석사)
1997년 영국 Univ of Edinburg (이학박사)
1999~현재 서울시립대학교 전자전기컴퓨터공학과 정교수
E-mail : ywlee@uos.ac.kr