Journal of Digital Contents Society Vol. 14 No. 4 Dec. 2013(pp. 447-454)
http://dx.doi.org/10.9728/dcs.2013.14.4.447

하둡 기반 빅데이터 영상 처리를 통한 차량 이동경로 추적 시스템의 설계 및 구현

양성은*, 최창열**, 최황규***

요 약

수많은 CCTV에서 기록·보관되는 영상 데이터가 폭발적으로 증가하면서, 빅데이터 환경에 적합한 CCTV 영상 데이터의 처리와 응용이 큰 이슈가 되고 있다. 본 논문에서는 대규모 CCTV 영상 데이터를 하둡 기반으로 병렬처리하고, 이를 활용한 VRT(Vehicle Route Tracking) 시스템을 설계 구현한다. VRT 시스템은 대규모 차량 번호판 인식 시스템의 특성을 가지며, 구글 맵을 통해 특정 차량의 이동경로를 빠른 시간 내에 추적 가능케 한다. 그리고 VRT 시스템의 성능 평가를 위한 실험을 통하여 단일 PC와 하둡 환경에서 대규모 CCTV 영상 데이터의 번호판 인식 시간을 비교·분석한다.

키워드 : CCTV, 차량 번호판 인식 시스템, 빅데이터, 하둡, 구글 맵

Design and Implementation of Vehicle Route Tracking System using Hadoop-Based Bigdata Image Processing

Seongeun Yang*, Changyeol Choi**, Hwangkyu Choi***

Abstract

As the surveillance CCTVs are increasing every year, big data image processing for the CCTV image data has become a hot issue. In this paper, we propose a Hadoop-based big data image processing technique to recognize a vehicle number from a large amount of automatic number plate images taken from CCTVs. We also implement the vehicle route tracking system that displays the moving path of the searched vehicle on Google Maps with the related information together. In order to evaluate the performance we compare and analysis the vehicle number recognition time for a lot of CCTV image data in Hadoop and the single PC environment.

Keywords : ANPR, Big Data, CCTV, Google Maps, Hadoop

1. 서론

공공기관에서 범죄 예방과 치안 강화 등의 목적으로 설치하는 CCTV의 활용성이 높아지면서, CCTV 대수가 급증하고 있다. 통계청 자료에 의하면 2012년 공공기관 CCTV 설치 대수는 총 452,725대로, 전년도에 비해 8만대 이상, 2009년에 비해 2배 정도 증가되었다[1]. CCTV가 늘어나면서 CCTV에서 수집하는 영상 데이터의 양도 폭발적으로 증가하여 점점 빅데이터로 되고 있다[2]. 하지만 아직까지 수많은 CCTV로 부터 수집되는 데이터를 빅데이터로 인식하는 활용 영역과 빠른 처리 기법을 찾는 연구는 드문 상황이다.

CCTV를 응용하는 차량 번호판 인식 기술[3,4,7,8]은 지능형 교통 시스템의 핵심 기술로서 차량 번호판 인식 시스템에 결합된다. 차량 번호판 인식 시스템은 CCTV나 카메라에서 촬영한

※ 교신저자(Corresponding Author): Hwangkyu Choi
접수일: 2013년 11월 7일, 수정일: 2013년 12월 07일
완료일: 2013년 12월 18일
* 강원대학교 컴퓨터정보통신공학전공
Tel: +82-2-880-7008 , Fax: +82-2-880-7010
email : sey@kangwon.ac.kr
** 강원대학교 컴퓨터정보통신공학전공
*** 강원대학교 컴퓨터정보통신공학전공

차량의 영상 데이터를 인식하고 차량 번호를 문자로 출력 또는 저장한다. 주로 주차 관제 서비스나 실시간 범죄 차량 검거 등에 사용되며, 데이터의 실시간 처리가 요구된다. 하지만, 차량 번호판 인식 시스템의 실시간 처리 방식으로 대규모 데이터를 일괄 처리하는 경우에 데이터의 처리 시간이 오래 걸리고, 처리할 데이터의 양도 너무 많다. 이러한 차량 번호판 인식 시스템의 실시간 처리의 요구 특성을 충족시키기 위해서 많은 양의 데이터를 여러 노드로 분산하고, 이를 동시에 병렬 처리하는 빅데이터 처리 기술의 적용이 필요하다.

빅데이터 처리 기술로는 하둡(Hadoop), No SQL, R 등이 있으며, 그 중에서도 가장 대표적인 하둡은 현재 정형 및 비정형 빅데이터 분석에서 가장 선호되는 솔루션이다. HDFS(Hadoop Distributed File System)에 파일을 저장하고, 맵-리듀스(Map-Reduce) 방식으로 많은 양의 데이터를 빠른 속도로 분산 병렬 처리한다[5,6]. 현재 야후와 페이스북 등에서도 사용하고 있으며, 채택하는 회사가 늘어나고 있다.

본 논문에서는 차량 번호판 인식 시스템의 실시간 처리 방식을 보완하는 일괄 처리 방식의 하둡 기반 VRT(Vehicle Route Tracking) 시스템 설계를 제안한다. 제안하는 VRT 시스템은 대규모 CCTV 영상 데이터를 빠른 시간 내에 차량 번호 문자로 출력하며, 출력된 차량 번호들 중에서 특정 차량을 검색하여 해당 차량의 번호와 관련 정보(촬영 장소와 일시)를 함께 결과 파일에 기록한다. 결과 파일의 차량 정보는 여러 응용 시스템에서 활용이 가능해지며, VRT 시스템은 검색한 차량의 정보를 구글 맵에 표시하여 차량의 이동경로를 추적 할 수 있게 한다. 그리고 VRT 시스템의 성능으로서 단일 PC와 하둡 환경에서 번호판 인식 시간을 측정, 비교한다.

본 논문의 구성은 다음과 같다. 2장에서는 차량 번호판 인식 기술, 하둡, 구글 맵 API 기술에 대해 소개하고, 3장에서는 시스템 구성, 동작 및 데이터 흐름, 구현 과정을 설명한다. 4장에서는 실험 환경, 실험 방법을 서술하고, 실험 결과를 비교·분석한다. 마지막으로 5장에서는 결론과 향후 과제를 다룬다.

2. 관련 연구

2.1 차량 번호판 인식 기술

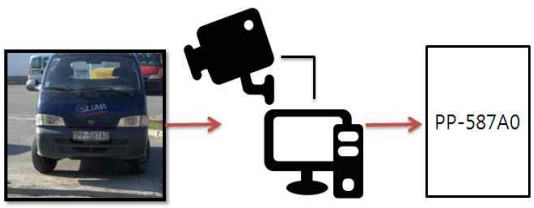
차량 번호판 인식 기술은 지능형 교통 시스템의 핵심 기술로서 ANPR(Automatic Number Plate Recognition)[3] 또는 LPR(License Plate Recognition)[4]이라고 부른다. 주로 주차 관제 서비스나 실시간 범죄 차량 검거 목적의 시스템에 사용되며, 카메라나 CCTV에 찍힌 차량의 번호판 영상을 인식하여 차량 번호를 문자로 출력 또는 저장하는 기술을 말한다. 현재까지도 처리 속도와 인식률을 높이기 위한 연구가 활발히 진행되고 있으며, 다양한 알고리즘이 존재한다[7,8].

2.1.1 차량 번호판 인식 시스템

대부분의 차량 번호판 인식 시스템은 차량의 영상 데이터를 실시간으로 처리하여 결과를 출력한다. 이러한 기존의 차량 번호판 인식 시스템은 대규모 데이터의 일괄 처리가 필요한 빅데이터 환경에는 적합하지 않다.

(그림 1)은 기존의 차량 번호판 인식 시스템의 실시간 처리 방식을 보여준다. 한 번에 하나의 차량 영상 데이터를 실시간으로 처리하여 하나의 차량번호를 문자로 출력하는 과정을 나타낸다.

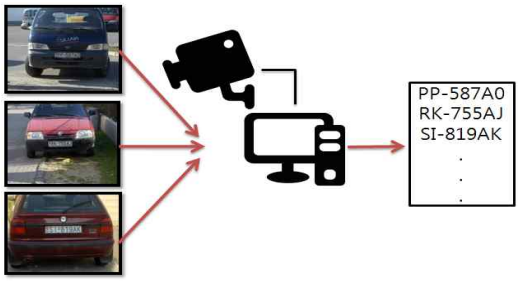
(그림 1) 기존의 차량 번호판 인식 시스템의 실시간 처리 방식



(Figure 1) The overall structure of the typical ANPR system

(그림 2)는 다수의 차량 영상 데이터를 기존의 차량 번호판 인식 시스템으로 처리하는 경우를 보여준다. 기존의 차량 번호판 인식 시스템으로 다수의 차량 영상 데이터를 처리하면, 처리 시간이 영상 데이터 개수만큼 증가된다. 이러한 처리 방식으로 대규모 영상 데이터를 처리하기에는 처리 시간이 오래 걸리고, 처리 할 수 있는 데이터 량의 한계가 있어 보인다.

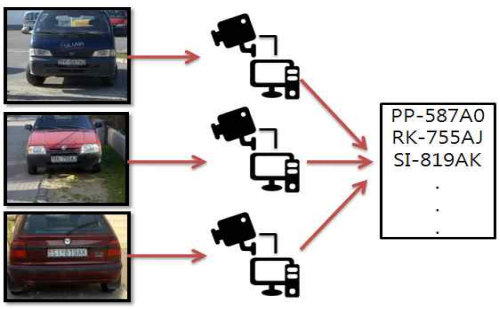
(그림 2) 다수의 차량 영상 데이터를 기존의 차량 번호판 인식 시스템으로 처리할 경우



(Figure 2) Bigdata recognizing procedure using the typical ANPR system

본 논문에서는 (그림 3)과 같은 분산 병렬 처리 방식을 제안한다. 다수의 차량 영상 데이터를 동시에 여러 개의 차량 번호판 인식 시스템으로 처리하여 빠른 시간 내에 차량 번호를 문자로 출력할 수가 있다.

(그림 3) 다수의 차량 영상 데이터를 제안하는 방식으로 처리할 경우



(Figure 3) Bigdata recognizing procedure using our proposed system

2.1.2 JavaANPR 소프트웨어[9]

JavaANPR 소프트웨어는 체코의 BRNO 대학에서 2007년에 제작된 자바 기반의 차량 번호판 인식 소프트웨어이며, 평균 85%의 문자 인식률을 갖는다. JavaANPR 소프트웨어는 오픈 소스이고, 자바 기반이기 때문에 하둡과 쉽게 결합이 가능하다. 본 논문에서는 JavaANPR 소프트웨어를 (그림 3)에서 제안한 방식으로, 동시에 다수의 차량 영상 데이터를 처리 할 수 있도록 설계한다.

2.2 하둡

하둡은 대용량 데이터를 분산 처리 할 수 있는 자바 기반의 오픈소스 프레임워크이며, 2005년에 더그 커팅(Doug Cutting)이 구글이 논문으로 발표한 GFS(Google File System)와 맵-리듀스를 구현한 결과물이다. 처음에는 오픈소스 검색 엔진인 너치(Nutch)에 적용하기 위해 시작됐다가 이후 독립적인 프로젝트로 만들어졌고, 2008년에는 아파치 최상위 프로젝트가 되었다[5,6]. 하둡은 분산 파일 시스템인 HDFS에 데이터를 저장하고, 분산 처리 시스템인 맵-리듀스를 이용해 데이터를 처리한다. 본 논문에서는 하둡 기반 시스템을 설계하여 CCTV 영상 데이터를 HDFS에 저장하고, 맵-리듀스를 통해 데이터를 분산 병렬 처리한다.

2.2.1 HDFS

HDFS(Hadoop Distributed File System)는 수십 테라비이트 또는 페타바이트 이상의 대용량 파일을 분산된 서버에 저장하고, 많은 클라이언트가 저장된 데이터를 빠르게 처리할 수 있게 설계된 파일 시스템이다[10]. HDFS에 저장하는 파일은 특정 사이즈의 블록으로 나눠서 분산된 서버에 저장된다. 기본적으로 하나의 블록은 64MB로 설정되어 있고, 복제 본을 3개씩 저장한다. 본 논문에서는 영상 데이터를 HDFS로 보내는 시간을 단축시키기 위해 복제 본을 1개씩 저장하도록 설계한다.

2.2.2 맵-리듀스

하둡의 맵-리듀스(Map-Reduce)는 방대한 양의 데이터를 병렬로 처리하게 해주는 프로그래밍 모델이다[11,12]. 맵-리듀스 프로그래밍 모델은 단순하게 맵(Map)과 리듀스(Reduce)라는 두 개의 메서드로 구성되고, 맵-리듀스 프레임워크를 이용하면 대규모 분산 컴퓨팅 혹은 단일 컴퓨팅 환경에서 개발자가 대량의 데이터를 병렬로 분석할 수가 있다. 입력 데이터는 여러 개의 매퍼로 분산된다. 매퍼는 데이터를 입력 받아 파티셔너로 보내고 Output Key, Value를 출력한다. 리듀서는 셔플을 통해 매퍼의 Output Key, Value를 Input Key, Value로 입력받는다. 리듀서에서 출력된 Output Key, Value는 출력 데이터로 병합되어 저장된다.

본 논문에서는 HDFS에 저장된 대규모 차량의 영상 데이터를 매퍼의 입력으로 받아 촬영

위치와 일시를 Output Key, 차량 번호를 Output Value로 출력한다. 리듀서는 매퍼의 Output Key와 Value를 입력받아 특정 차량의 번호를 검색하고, 다시 검색한 차량의 촬영 위치와 일시를 Output Key, 차량 번호를 Output Value로 출력하여 HDFS에 저장한다.

2.3 구글 맵 API

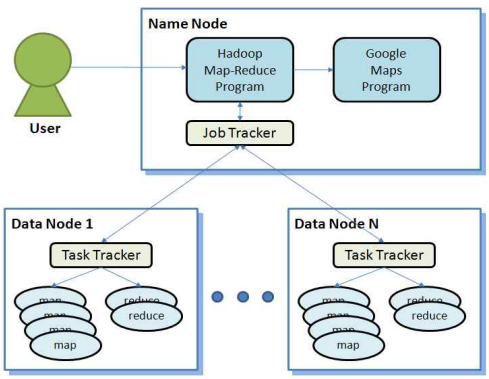
구글 맵은 구글(Google)에서 제공하는 기술로 웹 매핑(Web Mapping) 서비스 어플리케이션(Application)이라고 불린다. 구글에서는 구글 맵에 관한 다양한 구글 맵 API[13]를 제공하며, 개발자들은 이 API를 이용하여 매시업(Mashup)하거나 다양한 어플리케이션을 개발한다. 본 논문에서는 구글 맵 API를 이용하여 차량의 이동경로를 추적하는 VRT 시스템을 구현한다.

3. 시스템 설계 및 구현

3.1 시스템 구성

VRT 시스템은 (그림 4)와 같이, 하둡 맵-리듀스 프로그램과 구글 맵 프로그램, 잡 트래커와 태스크 트래커로 구성된다.

(그림 4) 제안된 VRT 시스템의 구성도



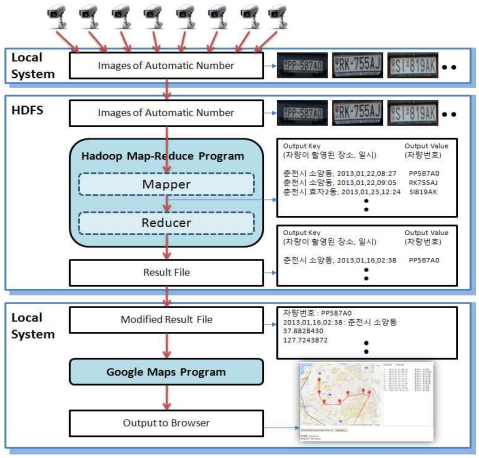
(Figure 4) The overall structure of the proposed VRT system

사용자가 네임 노드의 하둡 맵-리듀스 프로그램을 실행하면 잡 트래커는 여러 데이터 노드의 태스크 트래커로 일을 분배한다. 태스크 트래커는 하둡 맵-리듀스 프로그램에서 설정한 매퍼와 리듀서 개수만큼 맵과 리듀스를 실행하고, 처리 결과를 다시 잡 트래커로 보낸다. 잡 트래커가 일을 끝내면, 하둡 맵-리듀스 프로그램은 구글 맵 프로그램을 호출하여 사용자에게 최종 구현 화면을 보여준다.

3.2 동작 및 데이터 흐름

VRT 시스템의 동작 및 데이터 흐름은 (그림 5)와 같으며, CCTV의 영상 데이터는 미리 수집하였다고 가정한다. 작업 순서는 아래와 같다.

(그림 5) 제안된 VRT 시스템의 동작과정 및 데이터 흐름



(Figure 5) The functions and data flow of the proposed VRT system

1) CCTV에 저장 된 차량 번호판 영상 데이터를 로컬 시스템에서 수집한다.
2) 수집한 차량 번호판의 영상 데이터를 HDFS로 보낸다.
3) HDFS에서 하둡 맵-리듀스 프로그램을 실행시키면 매퍼가 먼저 동작한다. 매퍼는 JavaANPR 소프트웨어를 호출하여 차량 번호판 영상 데이터를 문자로 변환한다. 그리고 촬영 위치와 일시를 Output Key, 문자로 변환된 차량 번호를 Output Value로 설정하고 리듀서로 보낸다.
4) 리듀서는 매퍼로부터 받은 Output Key, Value를 Input Key, Value로 사용하여 하둡 맵-리듀스 프로그램에 입력된 특정 차량 번호를 검색한다. 검색된 차량 번호를 시간 순으로 정렬하고, 촬영 위치와 일시를

Output Key, 차량 번호를 Ouput Value로 설정하여 HDFS에 결과 파일로 저장한다.

5) 리듀서의 작업이 끝나면 하둡 맵-리듀스 프로그램은 LocationTransform 클래스를 호출한다. 이 클래스는 HDFS에 저장된 결과 파일 내용 중에서, 촬영 장소를 위도와 경도 좌표로 수정한다. 차량 번호와 촬영 일시를 수정한 좌표와 함께 로컬 시스템에 결과 파일로 저장한다.
6) 저장이 완료되면 LocationTransform 클래스는 로컬 시스템에 있는 구글 맵 프로그램을 호출한다.
7) 구글 맵 프로그램은 웹 브라우저를 호출하고, 웹 브라우저가 호출 되면 사용자가 직접 로컬 시스템의 결과 파일을 입력한다. 사용자의 입력이 끝나면, 구글 맵 프로그램은 위도, 경도 좌표를 구글 맵에 표시하고, 각 위치를 선으로 연결한다.

3.3 구현

3.3.1 하둡 맵-리듀스 프로그램

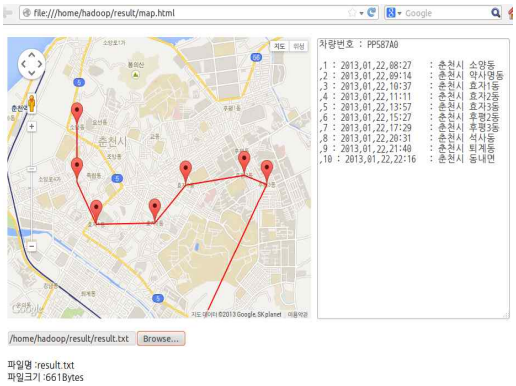
하둡 맵-리듀스 프로그램은 크게 Job, Mapper, Reducer 클래스로 구성된다. Job 클래스에서는 하둡 프로그램의 환경설정을 하고 Mapper와 Reducer작업이 끝난 후에는 촬영 장소를 위도, 경도 좌표로 변환하는 LocationTranform 클래스를 호출한다. Mapper 클래스는 javaanpr 클래스를 호출하여 차량 번호판 영상을 차량 번호 문자로 변환한다. 그리고 영상 데이터가 저장된 폴더 이름을 차량의 촬영 장소로 읽어오며, 촬영 일시는 랜덤으로 생성한다. 문자로 변환된 차량 번호, 촬영 장소와 일시는 Reducer 클래스로 보내진다. Reducer 클래스는 Mapper 클래스에서 보낸 차량 번호들 중에서 특정 차량 번호를 검색하여 차량에 관한 정보(차량 번호, 촬영 장소와 일시)를 HDFS에 기록한다. ImageInputFormat 클래스는 Job 클래스에서 입력받는 데이터의 형태를 정의하며, PathReader 클래스를 상속하였다. PathReader 클래스는 HDFS의 영상 데이터가 저장된 폴더의 경로를 읽어서 각각의 Mapper 클래스에 할당한다.

3.3.2 구글 맵 프로그램

하둡 맵-리듀스 프로그램의 실행이 끝나면 구글 맵 프로그램이 실행된다. 구글 맵 프로그램의 동작 과정은 아래와 같다.

1) initialize() 함수를 호출하여 설정된 좌표를 시작 위치로 지정하고 웹 브라우저에 구글 맵을 출력한다. 입력 파일이 있으면 readFile() 함수를 호출하고, 없으면 구글 맵만 출력한다.
2) readFile() 함수는 결과 파일을 읽어서 촬영된 위치의 좌표를 addMarker()로 보내고, 차량 번호와 촬영된 위치, 촬영 일시를 저장한다.
3) addMarker() 함수는 좌표를 구글 맵에 표시하고 drawPloyline() 함수를 호출한다.
4) drawPolyline() 함수는 좌표를 선으로 연결한다. 마지막으로 readFile() 함수에서 저장한 차량 번호, 촬영된 위치, 촬영 일시 정보를 리스트로 출력한다. 입력 파일의 이름과 크기 정보도 같이 출력한다.

(그림 6) 구현된 VRT 시스템 화면의 예



(Figure 6) An example display of the implemented VRT system

(그림 6)은 VRT 시스템 구현 화면의 한 예로써, 구글 맵 프로그램의 실행 화면과 동일하다. 이 화면에서는 춘천시에서 2013년 1월 22일 하루 동안 촬영 한 CCTV 영상 데이터를 하둡 맵-리듀스 프로그램으로 처리한 결과 파일을 입력으로 사용하였고, 검색한 차량 번호는 'PP587A0'이다. 이 차량이 처음 발견 된 장소는 춘천시 소양동이고 촬영 된 시간은 8시 27분이다. 그 이후, 이 차량은 약사명동, 효자1동, 효자2동, 효자3동, 후평2동, 후평3동, 석사동, 퇴계동, 동내면을 순서대로 지나갔다. 오른쪽 화면에는 차량의 이동 내역을 보여준다. 검색한 차량의 번호. 촬영 일시, 촬영 위치를 리스트로 출력한다. 왼쪽 화면에는 구글 맵에 차량의 이동 경로를 표시한다. 입력 파일의 좌표를 읽어서 시간 순으로 연결한

다. 아래 화면에는 입력 파일의 정보를 출력한다. 입력 파일의 이름은 “result.txt”이고 파일의 크기는 661바이트이다. (그림 6)과 같이 구글 맵으로 차량의 이동경로를 한 눈에 파악할 수 있을 뿐만 아니라, 구글 맵의 확대, 축소, 위성 기능 등을 활용하면 더욱 현실적인 위치를 확인할 수 있다.

4. 실험 및 분석

4.1 실험 환경

단일 PC와 하둡 환경에서의 PC 사양은 모두 동일하며, CPU는 Intel(R) Core(TM) i5-3470 CPU @ 3.2GHZ, RAM은 6GB, HDD는 2TB, OS는 Ubuntu 12.04 LTS(64비트)이다. 하둡은 안정적인 1.0.4버전을 사용하였고, 자바는 최신 버전인 JDK 1.7버전을 사용하였다. 실험 데이터로는 100KB 미만의 차량 번호판 정지 영상을 12만 8천개(6.6GB), 25만 6천개(13.2GB), 128만개(66GB), 256만개(132GB)를 생성하여 사용하였다.

4.2 실험 방법

4.2.1 단일 PC

단일 PC에서는 JavaANPR 소프트웨어의 GUI 부분을 제거하고, 실험 데이터 개수만큼 차량의 영상 데이터 인식 과정을 반복 실행하여, 차량 번호를 텍스트(text) 파일에 문자로 기록하는 프로그램을 구현하였다. 그리고 반복문이 끝나는 시간과 시작 전 시간의 차를 구하여 수행 시간을 측정하였다.

4.2.2 하둡 환경

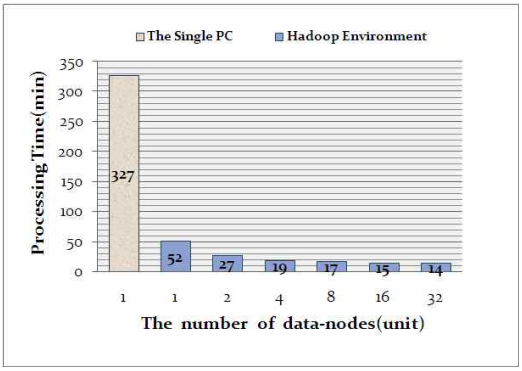
하둡 환경에서는 총 34대의 PC를 사용하였고, 네임 노드 1대, 보조 네임 노드 1대, 데이터 노드는 최소 1대부터 최대 32대까지 구성하였다. JavaANPR 소프트웨어가 결합된 VRT 시스템의 하둡 맵-리듀스 프로그램의 수행 시간을 측정하였다.

4.3 결과 분석

4.3.1 데이터 노드 개수에 따른 수행시간

(그림 7)은 실험 데이터가 12만 8천개(6.6GB)일 때, 단일 PC와 하둡 환경의 데이터 노드 개수에 따른 수행 시간을 보여준다.

(그림 7) 단일 PC와 하둡 환경에서의 데이터 노드 개수에 따른 수행 시간 비교 (6.6GB)

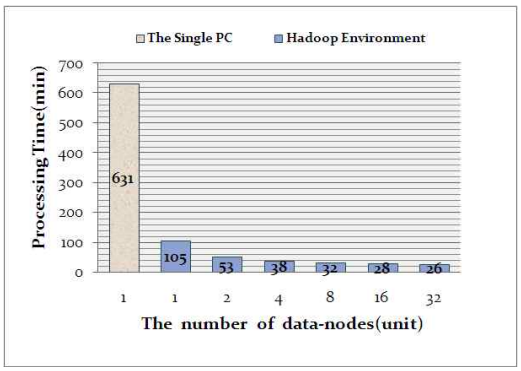


(Figure 7) Comparison of processing times varying the number of data-nodes under the Hadoop and single PC environment (6.6GB)

단일 PC의 수행 시간은 약 327분이다. 한 개의 차량 영상 데이터를 인식하는데 0.1533초가 걸리며, 초당 6.5장의 차량 영상 데이터를 인식할 수가 있다. 하둡 환경에서는 데이터 노드가 1대 일 때, 수행 시간은 약 52분이다. 한 개의 차량 영상 데이터를 인식하는데 0.0244초가 걸리며, 초당 41.0개의 차량 영상 데이터를 인식 할 수가 있다. 그리고 데이터 노드가 32대 일 때, 수행 시간은 약 14분이다. 한 개의 차량 영상 데이터를 인식하는데 0.0065초가 걸리며, 초당 152.3장의 차량 영상 데이터를 인식 할 수가 있다. 이는 단일 PC의 번호판 인식 속도의 23.4배에 해당한다.

(그림 8)은 (그림 7)의 실험 데이터를 2배로 늘렸을 때, 단일 PC와 하둡 환경의 데이터 노드 개수에 따른 수행 시간을 보여준다. 데이터 크기가 두 배이기 때문에 (그림 7)보다 수행시간이 대략적으로 2배씩 증가한 것을 알 수 있다. 단일 PC의 수행 시간은 631분이다. 한 개의 차량 영상 데이터를 인식하는데 0.1479초가 걸리며, 초당 6.8장의 차량 영상 데이터를 인식 할 수가 있다. 그리고 최대 데이터 노드가 32대 일 때, 수행 시간은 26분이다. 한 개의 차량 영상 데이터를 인식하는데 0.0060초가 걸리며, 초당 164.1장의 차량 영상 데이터를 인식 할 수가 있다. 이는 단일 PC의 번호판 인식 속도의 24.1배에 해당한다.

(그림 8) 단일 PC와 하둡 환경에서의 데이터 노드 개수에 따른 수행 시간 비교(13.2GB)

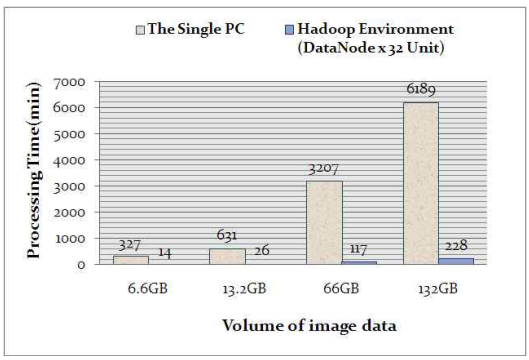


(Figure 8) Comparison of processing times varying the number of data-nodes under the Hadoop and single PC environment (13.2GB)

4.3.2 데이터 크기에 따른 수행시간

(그림 9)는 단일 PC와 하둡 환경의 데이터 노드 32대에서 수행 시간을 보여준다.

(그림 9) 단일 PC와 하둡 환경에서의 영상 데이터 크기에 따른 수행 시간 비교 (데이터노드 32대)



(Figure 9) Comparison of processing times varying the volume of image data under the Hadoop and single PC environment (32 units of data-node)

6.6GB와 13.2GB의 실험 데이터를 각각 10배씩 늘려서 수행 시간을 측정, 비교하였다. 실험 데이터가 66 GB와 132GB 일 때, 번호판 인식 속도는 단일 PC보다 각각 27.4배, 27.2배 빠르게 나타났다. 또한, 실험 데이터가 132GB 일 때, 하둡 환경에서 단일 PC 보다 수행 시간이 5961분 정도 빠르게 측정되었으며, 이는 시간으로는 약 99시간, 일수로는 약 4일 정도이다. 이 때, VRT 시스템은 최대 처리 속도를 가지며, 초당 187개의 번호판 영상 데이터를 인식 할 수가 있다.

5. 결론

대부분의 차량 번호판 인식 시스템이 요구하는 실시간 처리의 특성은, 한꺼번에 많은 양의 영상 데이터를 처리해야하는 빅데이터 환경에는 적합하지 않다.

본 논문에서는 차량 번호판 인식 시스템의 실시간 처리의 요구 사항을 충족시키기 위한 하둡 기반 VRT 시스템을 설계 및 구현하였다. 하둡 맵-리듀스 프로그램을 통해 HDFS에 분산 저장된 대규모 CCTV 차량 영상 데이터를 여러 노드에서 동시에 인식하여 차량 번호를 빠르게 출력하고 유용한 차량 정보로 가공하였다.

또한, 본 논문에서는 가공된 차량 정보를 입력으로 활용하여 특정 차량의 이동경로를 추적 할 수 있는 구글 맵 프로그램을 구현하였다. 구글 맵의 다양한 기능을 사용하여 더욱 자세한 위치를 확인 할 수도 있었다. 그리고 VRT 시스템의 성능 실험에서는 하둡 환경의 데이터 노드 개수가 32대일 때, VRT 시스템의 처리 속도가 단일 PC보다 최대 27배 빠르게 측정되었으며, 초당 187개의 번호판을 인식 할 수 있었다.

JavaANPR 소프트웨어를 우리나라 번호판 특성에 맞게 설계, 시험하는 것은 향후 과제로 남긴다.

References

[1] CCTV Installation and Operation of public institutions(2013), http://www.index.go.kr/egams/stts/jsp/potal/stts/PO_STTS_IdxMain.jsp?idx_cd=2855.

[2] C. Y. Jung, J. W. Han, J. S. Jang, “Big Data issues in video surveillance technology,” KIIT, vol. 10, no. 3, pp. 31-37, 2012.

[3] Qadri M.T, Asif M, “Automatic Number Plate Recognition System for Vehicle Identification Using Optical Character Recognition”, Education

Technology and Computer, pp. 335-338, 2009.

[4] Christos-Nikolaos E. Anagnostopoulos, Ioannis E. Anagnostopoulos, Ioannis D. Psoroulas, Vassili Loumos, Eleftherios Kayafas, "License Plate Recognition From Still Images and Video Sequences: A Survey", IEEE Transactions on Intelligent Transportation Systems, vol. 9, no. 3, pp. 377-391, Sep. 2008.

[5] J. H. Jung, "Beginning Hadoop Programming Development and Operations", Wikibooks Press, 2013.

[6] Tom White, "Hadoop: The Definitive Guide", 2nd ed., O'Reilly, Sebastopol, CA, 2011.

[7] Shih-Jui Yang, Ho, C.C, Jian-Yuan Chen, Chuan-Yu Chang, "Practical Homography-based perspective correction method for License Plate Recognition", Information Security and Intelligence Control (ISIC), pp. 198-201, 2012.

[8] S. Lew, S. Y. Choi, W. J. Lee, B. R. Lee, K. W. Min, H. C. Kang, "Extraction of the License Plate Region Using HoG and AdaBoost", Journal of Digital Contents Society, vol. 10, no. 4, 2009.

[9] Ondrej Martinsky, "Algorithmic and Mathematical Principles of Automatic Number Plate Recognition Systems", Brno University of Technology, 2007.

[10] K. Shvachko, H. Kuang, S. Radia, R. Chansler, "The Hadoop Distributed File System," 26th IEEE Symposium on Mass Storage Systems and technologies, Yahoo!, Sunnyvale, pp. 1-10, May. 2010.

[11] J. Dean, S. Ghemawat, "Mapreduce: Simplified Data Processing on Large Clusters," 6th Symposium on Operating Systems Design and Implementation, Berkeley, USA, Dec. 2004.

[12] J. H. Chung, "Design of Trajectory Data Indexing and Query Processing for Real-Time LBS in MapReduce Environments", Journal of Digital Contents Society, vol. 14, no. 3, pp. 313-321, 2013.

[13] Google Maps API, http://code.google.com/intl/ko/apis/maps/documenta tion/javascript/.

양 성 은

2012년 : 강원대학교 컴퓨터정보통신공학전공(학사)
2012년 ~ 현 재 : 강원대학교 대학원 컴퓨터정보통신공학과 석사과정
관심분야 : 웹프로그래밍, 모바일컴퓨팅, 클라우드컴퓨팅, 데이터베이스시스템

최 창 열

1979년 : 경북대학교 전자공학과 (학사)
1981년 : 경북대학교 전자공학과 (공학석사)
1995년 : 서울대학교 컴퓨터공학과 (공학박사)
1984년 ~ 1996년 : ETRI 컴퓨터연구단 책임연구원/연구실장
1996년 ~ 현 재 : 강원대학교 IT대학 컴퓨터정보통신공학전공 교수
관심분야 : 컴퓨터아키텍쳐, 임베디드시스템, 모바일컴퓨팅

최 황 규

1984년 : 경북대학교 전자공학과 (학사)
1986년 : KAIST 전기및전자공학과 (공학석사)
1989년 : KAIST 전기및전자공학과 (공학박사)
1990년 ~ 현 재 : 강원대학교 IT대학 컴퓨터정보통신공학전공 교수
관심분야 : 데이터베이스시스템, 멀티미디어시스템, 클라우드컴퓨팅