KOREAN SOCIETY FOR INTERNET INFORMATION
사단법인 한국인터넷정보학회
EIRIC Electronic & Information Research Information Center 전자정보연구정보센터

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

루씬을 이용한 빅데이터 인덱싱 및 검색시스템의 설계 및 구현☆

A Design and Development of Big Data Indexing and Search System using Lucene

김 동 민[1] 최 진 우[1] 우 종 우[1*]
DongMin Kim JinWoo Choi ChongWoo Woo

요 약

최근 소셜 미디어 사용의 증가, 산업간 융합의 확대, 다양한 스마트 기기의 보급을 통한 인터넷의 이용이 증가하면서 수많은 데이터를 발생시키고 있다. 이들 데이터들은 크기가 매우 크고, 형식이 다양하며, 순환속도가 매우 빨라 기존의 데이터 처리기술만으로는 관리와 분석이 어려운 실정이다. 즉, 수십 테라에 이르는 데이터의 폭증 및 데이터의 다양화에 따라 빠르게 분석하는 기술이 미흡하며, 이러한 문제점들을 해결하기 위한 새로운 기술적 방안이 절실히 요구되고 있다. 이러한 빅데이터의 처리기술에 대한 많은 연구가 최근 활성화 되고 있으며, 본 연구에서는 이러한 관점에서 빅데이터 플랫폼의 효과적인 인덱싱 엔진의 설계 및 구현에 관하여 기술한다. 즉, 기존의 데이터 처리기술의 범위를 초과하는 대규모의 데이터 집합을 효율적으로 관리하고, 인덱싱을 통한 검색속도의 향상으로 데이터 분석 시 소요되는 시간 단축을 연구목표로 한다. 본 연구의 실험을 위해서는 대규모 SNMP(Simple Network Management Prtocool) 로그 데이터를 사용하였으며, 효율적 데이터의 인덱싱을 통한 빠른 검색으로 데이터 분석시의 시간을 최대한 단축하고자 하였다. 또한 분석된 데이터의 표현의 가시화를 통하여 사용자의 데이터 분석에도 도움이 될 것으로 기대한다.

☞ 주제어 : 빅데이터, 빅데이터 플랫폼, 데이터 인덱싱, 데이터 검색

ABSTRACT

Recently, increased use of the internet resulted in generation of large and diverse types of data due to increased use of social media, expansion of a convergence of among industries, use of the various smart device. We are facing difficulties to manage and analyze the data using previous data processing techniques since the volume of the data is huge, form of the data varies and evolves rapidly. In other words, we need to study a new approach to solve such problems. Many approaches are being studied on this issue, and we are describing an effective design and development to build indexing engine of big data platform. Our goal is to build a system that could effectively manage for huge data set which exceeds previous data processing range, and that could reduce data analysis time. We used large SNMP log data for an experiment, and tried to reduce data analysis time through the fast indexing and searching approach. Also, we expect our approach could help analyzing the user data through visualization of the analyzed data expression.

☞ keyword : Big data, Big data Platform, Data Indexing, Data Searching,

1. 서 론

최근 ICT의 최대 이슈들 중 하나인 빅데이터는 Gartner, ICS등 글로벌 ICT리서치 업체들이 ICT산업에 미칠 기술요소로 빅데이터를 선정하면서 관련산업에 대한 관심이 급증하고 있다[1]. 미국에서는 NSF, NASA등 29개의 정부부처가 참여하여 개방성과 상호호환성이 있는 빅 데이터 프레임워크 전략을 개발하고 있으며, 유럽에서도 여러국가의 연구기관이 다양한 분야의 데이터를 보유하기 위한 범 유럽 디지털 데이터 저장소 구축을 위한 노력을 하고 있다[2]. 이와 같이, 빅데이터는 소셜미디어, 산업간 융합의 확대, PC 및 스마트기기를 통한 인터넷의 이용이 급증하면서 대용량의 데이터를 발생시키고 있으며, 이들 데이터는 그 형식이 다양하고 순환속도가 매우 빨라서 기존의 데이터 처리 방식으로는 관리와 분석이 어려운 실정이다. 즉, 데이터의 폭증 및 데이터 다양화에 따라 처리속도를 요구하는 데이터 분석이 필요하나, 고개 요구를 만족 시킬 수 있는 분석기술이 미흡하며, 대용량의 데이터를 빠르게 처리하고 분석할 수 있는 기술 또

[1] School of Computer Science, Kookmin University, Seoul, 136-702, Korea.
* Corresponding author (cwwoo@kookmin.ac..kr)
[Received 1 October 2014, Reviewed 6 October 2014, Accepted 14 October 2014]
☆ 본 연구는 국민대학교 교내연구비의 지원으로 수행되었음

한 미흡한 실정으로 이들 문제점들의 해결을 위한 기술적 방안이 절실히 필요한 실정이다.

본 연구에서는 이와 같이 활성화 되고 있는 빅데이터 플랫폼의 인덱싱 엔진의 설계 및 구현에 관하여 기술한다. 즉, 기존의 데이터 처리기술의 범위를 초과하는 대규모의 데이터집합(정형 또는 비정형)을 효율적으로 관리하고, 이를 통한 검색속도의 증대를 위한 시스템의 구축을 목표로 한다. 또한 다양한 영역과의 연동 및 다각적 분석이 가능하도록 다음과 같은 연구목적으로 접근한다.

첫째, 기존의 빅데이터 처리를 위한 오픈소스 들을 분석하고, 이를 통해 기존 검색시스템들의 장단점을 찾아 설계에 반영함으로서 기존 시스템의 단점들을 개선하고자 한다. 둘째, 플랫폼 독립성을 위해 순수 자바 라이브러리만을 사용하여 구현하며, 객체지향기반 프로그래밍 방법론을 충실히 한다. 셋째, 빅데이터 처리 플랫폼과 비즈니스 로직, 즉, 데이터 처리에만 집중하여 개발한다. 넷째, 오픈소스 기반의 인덱싱 및 검색엔진은 기본적인 기능을 충실히 하며, 처리속도의 개선 및 기능의 확장시 이를 반영 할 수 있도록 확장성을 고려한다. 다섯째, 개발 시스템의 전체 성능향상을 위하여, 데이터 수집과 같은 초기 단계에서부터 분석 및 활용단계 까지 전반에 걸친 생명주기를 고려하여 설계 및 구현한다.

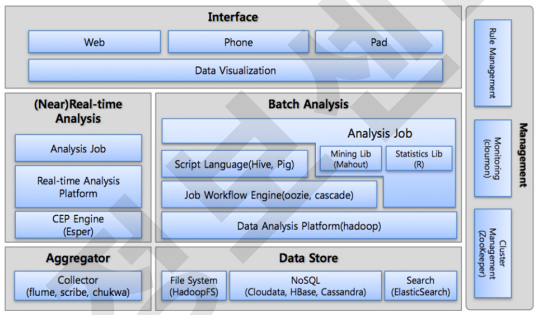
따라서 본연구의 결과인 효율적 인덱싱 및 검색 기법은 사용자에게 보다 높은 유연성을 제공할 수 있을 것이며, 컴포넌트 기반 설계로 인하여 높은 확장성을 제공할 수 있을 것이다. 또한 다양한 영역의 데이터 처리 및 분석뿐 아니라 향후 대규모 데이터 모델에서도 사용가능한 발전방향을 제시할 수 있을 것이다. 본 연구의 시험영역으로, 대규모 SNMP로그 데이터를 사용하여 의미 있는 정보의 발굴 및 지속적 모니터링을 통하여 장비의 결함을 신속히 찾아낼 수 있는 시스템 구축으로 하였다.

2. 관련 연구

2.1 관련 기술

2.1.1 빅데이터

빅데이터는 데이터의 부피가 크고 변화의 속도가 빠르며, 다양한 데이터 형식 및 속성을 가지는 데이터를 의미한다. 빅데이터 기술은 하둡(Hadoop)과 NoSQL의 성공을 기반으로 발전되고 있으며, 핵심기술인 하둡은 파일시스템과 분산처리에 초점을 두고 있는 플랫폼이지만 이를 기반으로 다양한 에코시스템이 구축됨으로서, 산업계 표준으로 진화되고 있다. 그림 1 에서와 같이 정보의 형태 및 표현구조가 상호 이질적인 다양한 구조(정형 및 비정형)의 특성을 가지는 데이터의 처리시에 요구되는 요소기술들을 하둡 에코시스템이라 한다. 이러한 요소기술들은 총 6 부분으로 구분되며, 이중 다수는 오픈소스의 형태로 일부 공개되어 있다.



(그림 1) 하둡 에코시스템의 소프트웨어 스택[3]
(Figure 1) Hadoop Echo System Software Stack

2.1.2 빅데이터 인덱싱

빅데이터 인덱싱 관련기술로는 루씬[4] 과 솔라[5]가 대표적이다. 루씬은 확장 가능한 고성능 정보검색 라이브러리로서, 자바로 구현되어 있고 아파치 소프트웨어 재단에서 아파치 소프트웨어 라이선스로 배포중이다. 루씬은 문서를 색인하고 색인된 문서를 검색하는 기능을 제공하며, 자료를 수집하거나 사용자로부터 질의를 받는 등의 기능은 제공하지 않는다. 따라서 루씬을 활용하기 위해서는 상기 기술된 기능들을 따로 구현하거나 해당기능을 제공하는 시스템과의 연동이 요구된다.

솔라는 매우 빠르고 활용도가 높은 오픈소스 엔터프라이즈 검색 플랫폼이다. 아파치 루씬 프로젝트에서 발전되었으며, 실시간 인덱싱과 다양한 포맷의 문서를 지원하며 풀 텍스트 검색등 강력한 기능을 제공한다. 솔라는 확장 가능하며 신뢰도가 높고 분산 인덱싱을 지원한다.

그 외, 아파치 주피커(Zookeeper)[6]는 공개 분산형 구성서비스, 동기서비스 및 대용량 분산시스템을 위한 네이밍 서비스를 제공한다. 파일시스템이나 트리 데이터구조와 비슷한 구조적 네임 스페이스 안에 데이터를 저장하고, 읽기 쓰기 작업을 수행함으로 분산 시스템을 구성한다.

2.2 빅 데이터 검색엔진

2.2.1 ElasticSearch[7]

셰이 배논(Shay Bannon)이 시작한 오픈소스 검색서버 프로젝트로, JSON 기반의 비정형 데이터 분산 검색과 분석을 지원한다. 이 검색엔진은 실시간 검색 서비스 지원과 분산 및 병렬처리 그리고 멀티테넌시(Multitenancy) 기능을 제공하며 다양한 기능을 플러그인(Plugin) 형태로 구현하여 적용할 수 있는 것이 특징이다. 또한 아마존 웹 서비스의 클라우드 서비스와 빅데이터 처리를 위한 하둡 연동도 지원하고 있다. ElasticSearch는 현재 웹 문서 검색, 소셜 데이터 분석, 쇼핑몰 검색등 다양한 서비스에서 사용되고 있으며, 광범위한 검색과 분석에 활용된다.

2.2.2 Lucandra/Solandra

Jake Luciani에 의해 소개된 Lucandra는 루씬의 백엔드를 카산드라[8] 기반으로한 인덱싱 및 검색 플랫폼이다[9]. 루씬의 인덱스 데이터를 NoSQL솔루션인 카산드라 데이터베이스에 저장함으로 인덱스 데이터를 효율적으로 관리하고 성능을 향상 시켰다. 그러나 Lucandra의 몇 가지 한계점 때문에 루씬 기반의 프로젝트를 솔라 기반으로 변경하면서 Solandra로 프로젝트가 변경되었다. 이로써 기존 Lucandra의 한계점이었던 대용량 질의와 UUID간의 매핑성능 문제들이 해결되면서 대용량 문서처리가 가능해 졌다.

2.3 빅데이터 인덱싱관련 연구

효율적 빅 데이터 인덱싱을 위한 연구는 다양한 관점에서 다수의 연구가 진행되고 있다.

2.3.1 내용기반 이미지 추출 시스템[10]

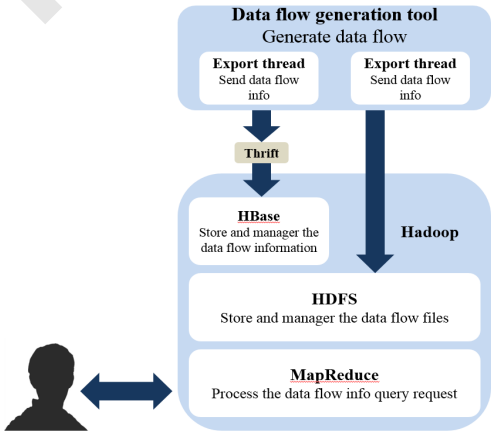
일반적으로 내용기반 이미지 검색시스템의 알고리즘은 시간복잡도가 매우 높다. 대규모의 데이터베이스에서 효율적인 이미지 검색을 위해서 하둡 분산 프레임워크이다. 이미지 검색시스템에서는 이미지의 특징점들을 추출하고, 이들을 관리하기 위한 해싱기법을 사용한다. 실험의 결과 처리시간은 데이터베이스에 저장된 이미지 특징의 수와 비례하며, 분산컴퓨팅과 내용기반 이미지 검색을 결합하여 대규모의 계산과 저장과 같은 몇 가지 문제를 효율적으로 해결 가능함을 제시하였다.

2.3.2 분산 실시간 텍스트 인덱싱 시스템[11]

분산 데이터 구조의 설계와 다수의 프로세서를 가지는 병렬시스템을 위한 분산 실시간 텍스트 인덱싱 알고리즘을 제시함으로, 로드 불균형의 감소와 확장 가능한 효율적 병합절차를 제시하고 있다. 시스템은 첫 단계에서 루씬의 인덱스를 세그먼트로 구성하고, 두 번째 단계에서 세그먼트의 병합을 수행하지만, 병목현상이 발생되기 때문에 확장성이 낮은 문제점이 있다. 따라서 시스템은 2-차원 해쉬 테이블을 구성함으로서 세그먼트의 병합 시 발생되는 병목현상을 해결하였고, 루씬의 방법보다 시간복잡도가 효율적임을 보였다.

2.3.3 하둡기반 데이터 처리시스템[12]

베이징 대학에서 연구한 빅데이터 관련 솔루션인 Hadoop-based Data Flow Management System은 기존 데이터 관리 시스템의 효율성 및 신뢰성과 확장성의 증대를 위해 그림2와 같이 MapReduce, HDFS, HBase로 구성하여 데이터 처리 및 관리를 분산, 병렬적으로 처리한다.



(그림 2) 하둡 기반 데이터 처리 시스템
(Figure 2) Hadoop based Data Processing system

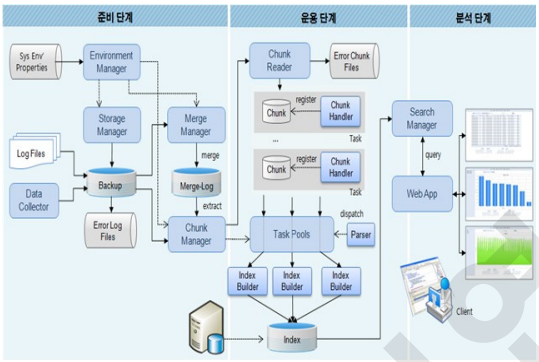
3. 시스템 설계

3.1 시스템 구성

본 연구에서는 대규모 데이터 처리를 위한 효율적 인덱싱 및 검색기술, 그리고 데이터 관계 표현기술 개발을 목표로 하며, 세부적으로는 다음 기능들의 달성을 목표

로 한다. 즉, 대규모 데이터의 인덱싱 관리기능, 사용자의 검색 질의 처리기능, 데이터 수집관리 기능, 저장소 관리 기능, 시스템 환경관리 기능, 그리고 사용자와의 상호작용을 담당하는 웹 응용 기능들의 개발을 포함한다.

이를 위하여 시스템은 그림 3과 같이 준비 단계, 운용단계, 분석단계의 3단계로 구성하였다. 첫째, 준비 단계에서는 다중 데이터들의 획득과 병합 및 구문분석과 더불어 데이터의 인덱싱 및 검색을 위한 준비 과정을 담당한다. 둘째, 운용단계에서는 사용자와의 상호작용을 수행하여 질의 검색에 따른 인덱싱 및 검색기능을 제공한다. 셋째, 분석단계에서는 사용자 질의에 대한 결과를 분석단계로서 질의 결과는 사실대로 표현되며, 필요시 다양한 형태의 그래프들을 제공하여 분석에 도움을 제공한다.



(그림 3) 시스템 구성도
(Figure 3) Structure of the System

3.2 시스템 세부기능

시스템의 전반적인 흐름은 준비단계에서부터 시작하여 운용단계를 거쳐 분석단계로 진행되며, 각 단계별로 다음과 같이 핵심모듈들로 구성된다.

3.2.1 준비단계

준비단계는 시스템 환경관리기 (Environment Manager), 데이터 수집기(Data Collector), 저장소 관리자(Storage Manager), 3가지가 존재한다.

. *시스템 환경관리기:* 준비단계에서 시스템 운용을 위한 환경 속성을 구성한 구성요소들의 정보를 관리하며 관련 시스템의 정보요청이 있을 경우 해당정보를 전달하는 역할을 수행한다.

. *데이터 수집기:* 네트워크를 구성하는 네트워크 엔티티들 중 스위치/라우터 또는 서버에 해당하는 장비들로부터 네트워크 관리 프로토콜인 SNMP[13]를 사용하여 장비의 상태 등에 대한 정보를 수집하는 역할을 담당한다.

. *저장소 관리자:* 기존에 존재하는 대용량 로그 데이터들의 일괄 병합처리 작업 및 관리를 수행하는 역할을 담당하며, 인덱스 파일의 저장데이터를 관리한다. 병합작업 수행 시 각 로그파일의 끝에 구분자를 삽입하여 구문분석 시 각 로그파일을 구분할 수 있도록 한다.

3.2.2 운용단계

운용단계에는 구문분석기(Syntax parser), 인덱스 빌더(Index builder)가 존재하며, 세부 기능은 다음과 같다

. *구분 분석기:* 구문분석기는 *SNMP로그 데이터 분석기*와 *사용자 질의 분석기*로 구분된다. *SNMP로그 데이터 분석기*는 병합된 로그 파일에서 단일 로그 데이터 사이에 있는 구분자를 통해 각 단일 로그 데이터를 구분하고, 인덱싱될 정보를 추출하여 인덱스 빌더에 전달하는 역할을 수행한다. *사용자 질의 분석기*는 검색 시 사용자가 입력한 질의에 대한 구문분석을 수행하는 역할을 담당한다.

. *인덱스 빌더:* 인덱스 빌더는 루씬 기반의 인덱스 파일을 생성하는 모듈이며, 인덱스 생성 시 구문분석기에서 추출된 정보를 바탕으로 루씬 인덱스 파일과 저장 데이터 파일을 생성한다. 생성된 인덱스 파일과 저장데이터 파일은 효율적 관리를 위해 저장소 관리자에서 관리한다.

3.2.3 운용단계

분석단계에는 검색기 (Search Manager), 웹 응용(Web App)이 존재한다.

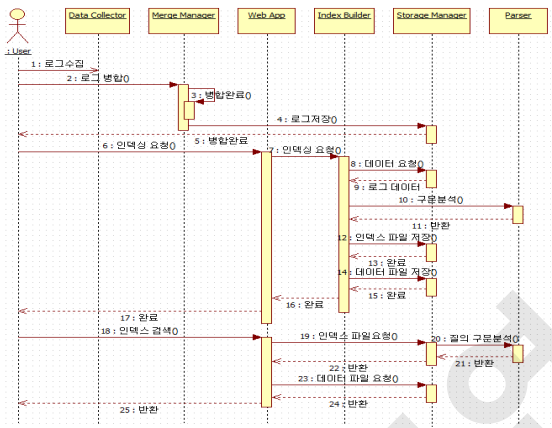
. *검색기:* 데이터 저장소에 저장된 인덱스 파일과 저장 데이터를 파일을 기반으로 사용자 질의에 맞는 데이터를 검색하여 반환하는 역할을 담당한다. 검색은 기본적으로 루씬의 질의 문법을 따르며, 해당 질의 구문분석은 구문분석기를 통하여 이루어진다. 분석된 질의를 이용해 인덱스 파일과 저장데이터 파일을 순회하여 사용자 질의 결과를 반환한다.

. *웹 응용:* 시스템의 사용 편의성을 위해 웹 인터페이스로 제작되며, 인덱싱 및 검색을 도와주는 역할을 담당한다. 또한 웹상에서 실시간으로 인덱싱 과정을 볼 수 있고, 사용자 질의 결과를 가시화 하여 사용자의 데이터 분석을 도와준다.

3.3 시스템 흐름

시스템 각 단계별 모듈의 주요 메시지 흐름은 그림4와 같으며, 메시지 흐름도에서 각 메시지들은 모듈간의 통신을 의미한다. 모듈의 메시지 흐름은 순차적이며 개략적으로 설명하면 다음과 같다.

사용자는 SNMP데이터 수집하여 수집된 데이터를 병합한다. 이후 사용자 요청에 의해 인덱싱이 이루어지면, 병합된 로그를 구문분석하여 인덱스 파일을 생성한다. 최종적으로 사용자의 검색요청에 의해 사용자 질의를 분석하고 인덱스 파일로부터 해당 데이터를 찾아 반환한다.



(그림 4) 시스템 흐름도
(Figure 4) Sequence diagram of the system

Message 1 : Collect SNMP Log Data
Message 2-3 : Merge Collected SNMP Log Data
Message 4-5 : Complete Merge Task
Message 6-7 : Request Indexing Process
Message 8-9 : Request Log Data and Return
Message 10-11 : Parse Log Data and Return
Message 12-13 : Create and Save Index File
Message 14-15 : Create and Save Data File
Message 16-17 : Complete Indexing Process
Message 18 : Request Searching Process
Message 19 : Request Index File
Message 20-21 : Parse User Query and Return
Message 22 : Return Index Field Information
Message 23 : Request Data File
Message 24 : Return Data Information
Message 25 : Complete Searching Process

4. 시스템 구현

4.1 시스템 구성

시스템에 대한 접근성 및 사용자 편의성을 제공하기 위해 본 시스템의 사용자 인터페이스는 웹 응용으로 개발하고, 개발언어로는 JAVA(JDK 1.7), HTML5, Javascript를 사용하였다.

4.1.1 SNMP 로그 데이터 수집기 구현

SNMP로그 데이터 수집기는 SNMP4J[14]를 사용하여 SNMP Agent 에 요청을 전송하고 응답 받는 기능을 수행한다. 설정에 따라 주기적으로 SNMP Agents에 요청을 전송하고 응답받은 메시지를 로그파일로 저장한다. SNMP 로그 데이터 수집을 위해 SNMP Agent가 설치된 서버는 클라이언트 PC에 가상 서버의 형태로 존재하며, 각 PC에 5대씩 가상 서버를 설치하여 SNMP 요청을 하여 메시지를 수신한다. 수신한 각 로그 데이터의 크기는 300KB를 넘지 않는다. SNMP 요청 주기는 분당 10건 이내로 하였으며 가상 서버 노드 당 20~40GB의 로그데이터를 수집한다. 그림 6은 수집기를 통해 수집된 SNMP 로그 데이터의 샘플 데이터이다. SNMP 로그 데이터는 그림 5와 같이 MIB(Management Information Base) 변수명과 MIB 변수 값의 쌍으로 되어 있다[15].

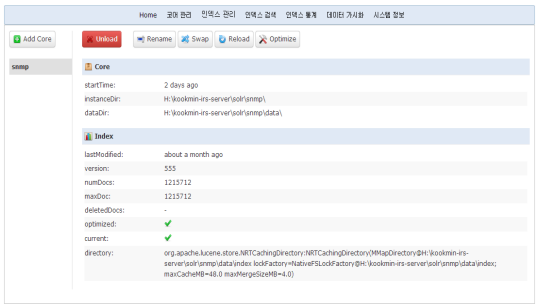
```
SNMPv2-MIB::sysORUpTime.8 = Timeticks: (57) 0:00:00.57
IF-MIB::ifNumber.0 = INTEGER: 2
IF-MIB::ifIndex.1 = INTEGER: 1
IF-MIB::ifDescr.1 = STRING: lo
IF-MIB::ifType.1 = INTEGER: softwareLoopback(24)
IF-MIB::ifMtu.1 = INTEGER: 65536
...중략...
SNMP-VIEW-BASED-ACM-MIB::vacmViewTreeFamilyStatus."_all_".1.0 = INTEGER: active(1)
SNMP-VIEW-BASED-ACM-MIB::vacmViewTreeFamilyStatus."_none_".1.2 = INTEGER: active(1)
SNMP-VIEW-BASED-ACM-MIB::vacmViewTreeFamilyStatus."_none_".1.2 = No more variables
left in this MIB View (It is past the end of the MIB tree)
```

(그림 5) SNMP 로그 샘플 데이터
(Figure 5) SNMP log sample data

4.1.2 웹 응용 구현

웹 응용은 사용자에게 인덱싱 및 검색기능의 사용 편의성과 데이터 가시화의 목적을 가진다. 기본적으로 솔라에서 제공하는 기능들을 흡수하고, 추가적으로 웹 인덱싱 및 검색, 데이터 가시화 기능을 구현한다. 추가적인 기능들은 JQuery기반 라이브러리로 구현하였으며 서버 사이드의 기능은 JAVA(JDK 1.7)로 작성되었다.

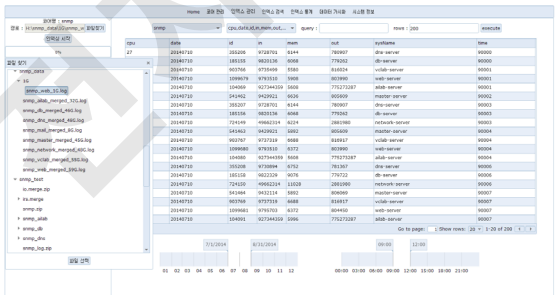
웹 응용은 크게 인덱스 관리, 인덱싱, 인덱스 검색, 인덱스 통계, 데이터 가시화의 기능을 제공한다. 그림 6은 인덱스 관리 화면이며, 화면의 구성은 솔라의 기능으로 구현되었다.



(그림 6) 인덱스 관리 화면
(Figure 6) Index Management Screen

인덱싱은 일괄 및 실시간 처리 모두를 지원하며 그림 7과 같이 웹 인터페이스에서 인덱싱할 로그 데이터를 선정하여 인덱싱 하고 인덱스 데이터를 검색하는 전 과정을 한 화면에서 볼 수 있도록 구성하였다. 파일 찾기를 통해 인덱싱 할 SNMP 로그 파일을 선택 후 시작 버튼을 누르면 진행바를 통해 인덱싱 진행상황을 실시간으로 확인 할 수 있다.

인덱스 데이터를 검색하기 위해서는 질의 상자에 질의문을 입력하고 실행 버튼을 선택한다. 검색 플랫폼은 솔라를 기반으로 사용하고 있으므로 검색 서버와의 통신은 솔라에서 전담하여 사용자 질의결과를 반환한다. 웹 응용에서는 반환받은 결과를 바탕으로 해당 데이터를 사용자가 보기 편하게 테이블 형식으로 보여준다. 그림 7은 웹 응용에서 인덱싱을 수행하고 해당 데이터를 검색하여 결과를 얻은 화면이다.



(그림 7) 웹 응용에서의 인덱싱 및 검색
(Figure 7) Indexing and Searching in Web App

데이터 가시화 기능은 인덱스 데이터의 시간흐름에 따른 변화를 보다 쉽게 보기 위한 기능으로 사용자 질의를 통해 특정 장비, 특정 시간대의 장비 정보를 살펴볼 수 있다. 그림 8은 시간흐름에 따른 CPU점유율의 변화량을 가시화한 그래프이다. 이를 통해 특정 시간대에 발생한 시스템의 이상 징후를 쉽게 발견할 수 있다.

인덱스 통계 기능은 장비별 통계 데이터의 가시화로 사용자의 데이터 분석을 지원 할 수 있다. 그림 9는 장비별 트래픽 통계가 그래프로 가시화된 화면이다. 장비별 트래픽 보기 메뉴를 선택하고 통계치를 볼 인덱스를 선택한 후 실행하면 해당 인덱스에 있는 전체 트래픽 데이터의 장비별 통계치를 가시화 하여 그래프로 표시한다. 해당 통계 데이터는 서버로의 요청 시 인덱스 데이터 집합에 있는 메타 데이터를 이용해 전체 트래픽 정보의 통계를 산출하고 해당 값을 장비별로 묶은 값을 기반으로 한다. 트래픽 통계 데이터를 가시화함으로써 언제 어떤 장비에 트래픽이 집중되고 있는지 보다 쉽게 알 수 있다.

(그림 8) CPU점유율의 변화
(Figure 8) Change of CPU occupancy

(그림 9) 트래픽 흐름 통계
(Figure 9) Statistics of Traffic flow

5. 실 험

5.1 시스템 환경

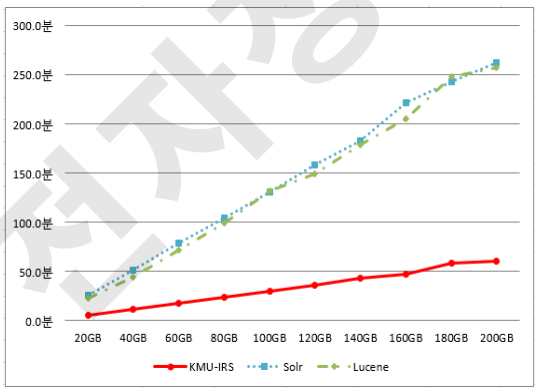
시스템의 실험 환경은 두 대의 IBM X3250M4 서버에 CentOS 6.3을 설치하여 본 시스템을 탑재 운용하였고, 하나의 L2 스위치와 두 대의 클라이언트 PC로 네트워크를 구성하였다. 두 인덱싱/검색 서버는 주키퍼(ZooKeeper)를 통해 분산 환경을 구축하였다.

5.2 인덱싱 성능분석

인덱싱 성능 분석은 데이터를 지속적으로 인덱싱 하고 검색하는 본 시스템의 특성상 가장 중요한 평가 요소라고 볼 수 있다. 인덱싱 성능 분석을 위해 인덱스 빌더 모듈을 분리하여 인덱스 작업의 시작점에서 시간을 측정하고 인덱스 파일이 생성되어 인덱스 작업이 끝난 시점에 시간차를 이용하여 인덱싱 속도를 측정하였다.

인덱싱 속도의 비교를 위해 루씬과 솔라의 인덱스 파일 생성 시간과 본연구의 시스템인 KMU-IRS와의 인덱스 파일 생성시간을 비교하였다. 시험 데이터는 20G에서부터 200G까지 20G씩 순차적으로 늘려가며 시험하였고, 해당 실험 결과는 그림 10과 같다.

실험결과 200G데이터를 인덱싱 할 경우 루씬과 솔라는 전체 데이터를 인덱싱 및 저장에 각 257분, 263분이 걸린 반면 본 시스템은 60.2분이 소요되었다. 실험적 데이터를 사용하였지만, 루씬과 솔라의 기본 알고리즘보다 상당히 향상된 결과를 보임을 알 수 있다.



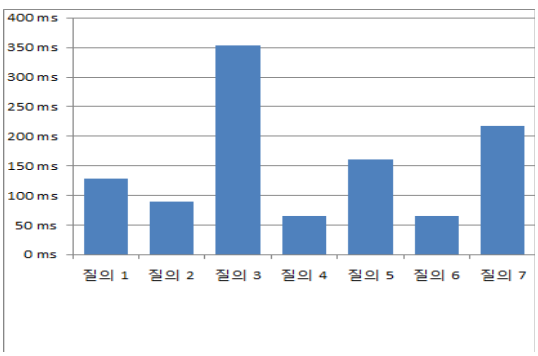
(그림 10) 인덱싱 성능 비교
(Figure 1) Comparison of Performance of Indexing

5.3 검색 성능 분석

본 시스템의 검색기는 루씬 인덱스 파일과 저장 데이터 파일을 기반으로 사용자 질의 결과를 반환한다. 검색 성능 역시 인덱스 성능 분석과 마찬가지로 사용자가 질의를 입력하고 검색을 수행하는 시점부터 최종 결과를 반환받는 시점까지의 시간을 측정하였다. 인덱스 파일의 검색은 루씬의 질의 문법을 따르며 성능 분석에 사용된 질의는 표 1과 같다. 그림 11은 표 1에 기술된 질의 문을 각 10번씩 수행한 결과의 평균을 나타낸 것이다. 질의 중 응답 속도가 가장 느린 결과는 353ms이다.

(표 1) 검색에 사용된 질의
(Table 1) Query used for Searching

질의 구분	질의 내용
질의 1	date:20140710 AND cpu:[* TO 10]
질의 2	date:20140710 OR cpu:[* TO 10]
질의 3	date:20140710 OR date:20140712 AND cpu:[* TO 10]
질의 4	date:20140710 OR date:20140712
질의 5	date:20140710 AND mem:6380 OR mem:5512
질의 6	date:20140710 AND cpu:[* TO 10] OR cpu:[90 TO *]
질의 7	date:20140710 AND cpu:[* TO 10] OR cpu:[90 TO *] OR mem:[6000 TO *]



(그림 11) 검색 소요시간
(Figure 11) Time for Searching

6. 결 론

본 논문에서는 기존의 데이터 분석 방식으로는 관리와 분석이 어려운 대용량 데이터에 대한 인덱싱 및 데이터 가시화를 도와주는 시스템을 설계 및 구현 하였다. 본 시스템은 데이터의 수집부터 인덱싱, 데이터 관리, 데이터 가시화를 한 시스템에서 통합적으로 관리하고 운용할 수 있는 점이 큰 특징이며, 사용자 인터페이스를 웹으로 구현함으로써 사용자의 접근성을 향상 시켰다. 또한 기존 루씬 인덱싱이 가졌던 대용량 데이터 인덱싱 및 저장 시의 성능 저하를 보완할 수 있는 저장 데이터 생성방식의 개선으로 성능을 향상 시켰다.

이번 실험에서는 정형적인 SNMP 로그 데이터를 인덱싱하고 검색하였지만, 다음 단계에서는 비정형 데이터의 경우에도 같은 결과가 나올 수 있는지 실험해 볼 것이다. 또한 현재 가시화부분의 기능이 제한적이므로 이를 보완하여 좀 더 다양한 형태의 데이터 가시화 기능을 제공하여 사용자의 데이터 분석을 도와주는 시스템으로 발전시키고, 전반적인 시스템의 완성도를 높이는 것이 향후 연구과제이다.

참 고 문 헌 (Reference)

[1] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. Byers, “Big data: The next frontier for innovation, competition, and productivity”, McKinsey Report, 2011.

[2] S. Lee, W. Sung, S. Park, “Future of Big Data Technology”, KOFST Issue Paper, 2012-03.

[3] B. Chung, H. Kim, W. Choi, “Future social and big data Technology”, IT Series. NIPA, 2012.

[4] E. Hatcher, O. Gospodnetić, and M. McCandless, “Lucene in action”, Manning Publications, Aug. 2010.

[5] R. Kuc, “Apache Solr 4 Cookbook”, Packt Publishing, Jan. 2013.

[6] F. Junqueira and B. Reed, “ZooKeeper : Distributed Process Coordination.”, O’Reilly Media, Inc., Nov. 2013.

[7] R. Kuc and M. Rogozinski, “ElasticSearch Server”, O’Reilly Media, Inc., Feb. 2013.

[8] Lakshman, Avinash, and P. Malik. ″Cassandra: a decentralized structured storage system.″ ACM SIGOPS Operating Systems Review 44.2 (2010): 35-40.

[9] J. Luciani, “Lucandra/Solandra: A Cassandra-based Lucene backend.”, http://blog.sematext.com/2010/02/09/lucandra-a-cassandra-based-lucene-backend/

[10] D. Yin and D. Liu, “Content-based Image Retrieval based on Hadoop”, Mathematical Problems in Engineering, Vol. 2013, Article ID 684615, (2013)

[11] A. Narang, V. Agarwal, M. Kedia, and V. Garg, “Highly Scalable Algorithm for Distributed Real-Time Text Indexing”, International Conference on High Performance Computing (HiPC), pp. 332－341 (2009)

[12] QIAO, Yuan-yuan, et al. ″Offline traffic analysis system based on Hadoop.″ The Journal of China Universities of Posts and Telecommunications 20.5 (2013): 97-103.

[13] D. Mauro and K. Schmidt, “Essential SNMP”, O’Reilly Media, Inc., Sep. 2005.

[14] Fock, Frank, and J. Katz. “SNMP4J-The Object Oriented SNMP API for Java Managers and Agents.”, http://snmp4j.org/index.html

[15] K. McCloghrie, M. Rose. “RFC 1066-Management Information Base for Network Management of TCP/IP-based Internets.”, TWG, Aug. 1988.

◐ 저 자 소 개 ◑

김 동 민 (DongMin Kim)
2014년 경북대학교 컴퓨터공학부 졸업(학사)
2014년~현재 국민대학교 대학원 컴퓨터공학과 석사과정
관심분야 : 인공지능, 시뮬레이션, 빅데이터, 웹 서비스.
E-mail : kdm1171@kookmin.ac.kr

최 진 우 (JinWoo Choi)
1998년 한성대학교 전산학과 졸업(학사)
2000년 국민대학교 대학원 전산과학과 졸업(석사)
2004년 국민대학교 대학원 전산과학과 졸업(박사)
2009년~현재 국민대학교 컴퓨터공학부 겸임교수
관심분야 : 인공지능, ITS, 지능형 에이전트, 정보 보호
E-mail : jnwochoi@kookmin.ac.kr

우 종 우 (ChongWoo Woo)
1991년 Illinois Institute of technology 전산학과 졸업(박사)
1994년~현재 국민대학교 컴퓨터공학부 교수
관심분야 : 지능형 에이전트, 상황인식, 모바일 게임, Modeling&Simulation
E-mail : cwwoo@kookmin..ac.kr