

과제명: 클라우드 컴퓨팅을 위한 기술 수요

과제 책임자: 우종욱

Jongwook Woo (jwoo5@calstatela.edu)

Associate Professor

Computer Information Systems Department

California State University Los Angeles

## 제출문

한국과학기술연구원장 귀하

이 보고서를 “클라우드 컴퓨팅을 위한 기술 수요” 과제의 최종 보고서로 제출합니다.

2011. 02. 09

주관연구기관명: California State University Los Angeles, HiPIC 연구센터

주관연구책임자: 우 종욱

## 요약문

1. 제목: 클라우드 컴퓨팅을 위한 기술 수요

2. 사업의 목적 및 필요성

본 기술조사는 멀티코어 기반 하드웨어 및 저가의 고용량 스토리지 인터넷 기술을 활용하여 IT 자원을 서비스로 제공하는 컴퓨팅 환경인 클라우드 컴퓨팅(Cloud Computing)을 위한 요구사항 및 구현을 위한 하드웨어 및 특히 소프트웨어 기술을 조사하는 것을 목적으로 한다.

웹 이전의 세계에서는 데이터가 부족해서 문제였으나 웹 이후 지난 16 여년간의 웹상에서의 데이터 축적은 데이터가 너무 많아서 문제이다. 왜냐하면 데이터 분석을 통하여 필요한 정보를 추출하는 시간과 비용이 너무 비싸지기 때문이다. 더구나 수많은 아날로그 정보를 디지털화 함으로써 축적된 방대한 데이터도 문제이고 아날로그 정보를 디지털화 하는 단순하지만 비싼 응용 시스템도 문제이다. 클라우드 컴퓨팅 특히 그 Map/Reduce 및 행 지향 데이터 베이스 방식은 이러한 방대한 데이터 중 단순하지만 비싼 데이터 추출 알고리즘을 병렬 계산 방식으로 쉽게 구현하는 해법을 제공한다.

덧붙여서 다음과 같은 사업 필요성을 요약 나열하였다.

- 인터넷 서비스 및 서비스운명을 위한 데이터가 매년 지속적으로 증가함에 따라 이들 대용량 데이터의 가공, 저장, 검색, 분석 등에 클라우드 컴퓨팅 활용이 증가되고 있음
- 대용량 데이터 처리를 위한 Google, Yahoo 등 글로벌 인터넷 서비스 업체의 클라우드 컴퓨팅 운영에 대한 기술 동향 분석이 필요함

- 지구온난화, 지진 예측 등 지질 및 천문기상 정보의 처리를 위한 표준기술로서 클라우드 컴퓨팅이 고려되고 있으며, 국내에서 해당 분야의 클라우드 컴퓨팅 활용 및 이에 따른 플랫폼 개발을 위해서는 해당 기술 동향 분석이 요구됨
- 클라우드 컴퓨팅 구축 시에 적용되고 있는 가상화 (Virtualization) 기술에 대한 세부기술의 특징 분석이 필요하며, 클라우드 컴퓨팅을 구성하는 자원에 대한 활용 시에 참조기술 분석이 필수적임

### 3. 사업의 내용 및 범위

본 기술조사에서는 클라우드 컴퓨팅을 위한 세부기술인 다음의 내용을 포함한다.

- 클라우드 컴퓨팅 응용 분야의 주요 기술 특징 및 동향 분석
- 물리적으로 분산되어진 스토리지의 논리적 구성을 위한 플랫폼 기술 동향
- 가상화를 통해 제공된 컴퓨팅 자원의 효율적인 처리를 위한 개발환경인 Map/Reduce 프로그래밍 기술 동향

### 4. 정보조사의 결과

대한민국의 LG, KT, SK등에서 클라우드 컴퓨팅 사이트를 개설하여 서비스중에 있으나 대용량 데이터의 저장이나 계산에 관한 서비스나 연구등은 산업체 학계 모두 미진하여 보인다. 본 보고서를 참조하여 소프트웨어의 선도 국가인 미국의 여러 시스템등을 예로써 보고 기존의 클라우드 컴퓨팅 기반을 이용하면 대한민국내의 대용량 데이터를 처리할 수 있는 기반이 구축될 수 있으리라 본다.

### 5. 정보조사의 활용계획

클라우드 컴퓨팅의 활용방안 및 기대효과는 다음과 같은 1) 구글에서 왜 Map/Reduce 방식이 고안되었는지와 2) 뉴욕타임즈에서의 예를 들어 설명하고자 한다. 덧붙여서 3) 아파치 하둡 프로젝트를 소개한다.

구글 연구원들이 당면했던 문제 중의 한 예는 수억의 사람들이 방문한 사이트들의 사용자 로그 정보에서 어떻게 하면 쉽고 빠르고 싸게 가장 방문빈도가 높은 사이트를 찾아 내어 방문빈도를 정렬하는가 하는 것이었다. 즉, 천문학적으로 방대한 데이터에서 비교적 쉬운 계산 알고리즘을 돌려서 빠르게 해법을 찾아내는 시스템의 구축이었다. 그래서 병렬 프로그래밍에서의 전통적인 Map/Reduce 알고리즘을 해법으로하는 시스템을 구글내에서 구축하게 된 것이다.

또한, 그 이후 미국에서는 전자상거래 사이트인 아마존에서 AWS 라는 아마존 클라우드 컴퓨팅 서비스를 구축하여 누구든지 싼 비용으로 클라우드 컴퓨팅을 이용하게 하였다. 이 서비스를 이용하면 비교적 단순한 프로그래밍 로직이지만 방대한 데이터를 가진 프로그래밍을 Map/Reduce 방식으로 돌려서 원하고자 하는 답을 찾아낼수 있다. 예를 들면, 2007 년도에 뉴욕타임즈에서 1851 년도 부터의 약 일천일백만개 되는 기사(약 4 테라 바이트 tiff 파일)을 pdf 파일 방식으로 바꿀려고 하였다. Tiff 파일을 pdf 파일로 바꾸는 아주 간단한 프로그래밍이지만 그 데이터 양이 천문학적으로 방대하여 그 비용과 시간이 너무 비쌌었다. 하지만 데렉이라는 소프트웨어 엔지니어가 아마존 AWS 의 EC2 와 S3 를 이용하여 단 \$240 로 24 시간내에 모두 pdf 파일로 바꾸어 냈다.

또한 아파치 하둡 [7, 12] 프로젝트는 이러한 Map/Reduce 병렬 계산 시스템을 구축할 수 있는 소프트웨어 플랫폼을 무료로 제공한다. 따라서 여러대의 컴퓨터 (하드웨어) 만 있으면 누구나 아마존의 AWS EC2 와 같은 Map/Reduce 시스템을 구축할 수 있다. 대한민국 정부, 연구소나 학교에서 Map/Reduce 시스템을 구축하는 프로젝트를 시발하고 Hadoop 을 이용한다면 Map/Reduce 소프트웨어를 처음부터 구현할 필요없이 Hadoop 을 그 시스템에 최적화 하는

일만 함으로써 대한민국 기업, 연구소, 기업체 및 일반인등을 위한 Map/Reduce 클라우드 컴퓨팅 시스템을 구현 할 수 있을것 이다.

본 기술 동향 조사서에는 앞서 서술한 예들을 구체적으로 나열하였고 미국에서의 클라우드 컴퓨팅 서비스 업체들 특히 Map/Reduce 서비스를 제공하는 업체들을 조사하고 그 특징을 서술하였다. 덧붙여서 이러한 대용량 데이터를 효율적으로 저장할 수 있는 행 지향 데이터 베이스에 대해 자세히 서술 하였다.

따라서 대한민국 내에서 그러한 서비스를 활용할 수 있는 힌트를 제시한다. 또한 아파치 하둡 및 행 지향 데이터 베이스에 대한 조사를 통하여 대한민국 내에서 하둡 및 행 지향 데이터 베이스 서비스를 설치가능한지에 대한 활용방안도 조사하고 서술하였다.

그리고, 미국내에서의 위와 같은 Map/Reduce 및 행 지향 데이터 베이스 시스템을 이용한 응용 예들을 분석하여 구체적으로 서술하였고 대한민국에서의 활용방안에 대한 예제를 제시하며 다음과 같은 기대효과를 나열한다. 즉, Map/Reduce 및 행 지향 데이터 베이스 클라우드 컴퓨팅 시스템을 통하여 방대한 데이터를 쉽고 빠르게 처리할 수 있는 기반이 구현되면 지질 및 천문기상 데이터를 분석하는데 이용 가능하며 웹 상에서는 사용자들의 웹 사용 행동 방식을 분석함으로써 사용자들 구미에 맞는 정보를 제공가능하다. 또한, 미국방부에서 연구는 것 처럼 방대한 웹사이트 데이터를 분석하여 적대 조직을 발견하거나 테러를 방지할 수 있는 시스템을 구현하는데 이용 할 수 있을 것이다.

## 목차

|                     |                               |
|---------------------|-------------------------------|
| 1. 서문               | 2                             |
| 2. 클라우드 컴퓨팅 소개      | 2                             |
| 2.1. 소프트웨어 기반       | 3                             |
| 2.2. 하드웨어 기반        | 4                             |
| 2.3. 클라우드 컴퓨팅의 유의점  | 4                             |
| 2.4. 클라우드 컴퓨팅의 장점   | 5                             |
| 3. 스토래지 가상화 플랫폼     | 6                             |
| 3.1. 스토래지 플랫폼       | 6                             |
| 3.2. 활용             | Error! Bookmark not defined.0 |
| 3.3. 동향             | Error! Bookmark not defined.1 |
| 4. MapReduce 컴퓨팅 자원 | Error! Bookmark not defined.1 |
| 4.1. 정의             | Error! Bookmark not defined.3 |
| 4.2. 아파치 하둡과 피그     | Error! Bookmark not defined.7 |
| 4.3. 산업체의 활용 동향     | Error! Bookmark not defined.8 |
| 5. 결론               | 21                            |
| 6. 참고 문헌            | 21                            |

## 1. 서문

인터넷과 웹의 발전으로 인하여 컴퓨터 통신 관련 응용 프로그램이 엄청나게 쏟아져 나왔다. 그러한 응용 프로그램을 이용함으로써 원격 컴퓨터 자원의 활용이 수월해 졌으며 원격 컴퓨터의 저장 및 계산 능력을 이용하는 비즈니스 모델이 각광을 받기 시작했다. 이러한 제 3자의 사업자들이 소유한 컴퓨터 소프트웨어 하드웨어 자원을 활용하기 위하여 그 사업자들의 컴퓨터 자원 활용 서비스를 이용하는것을 클라우드 컴퓨팅이라고 정의 한다. 즉 사업자들은 소유한 컴퓨터 자원을 고객들에게 제공하기 위하여 웹서비스등의 원격 조정 인터페이스등을 제공하고 고객들은 그 컴퓨터 자원을 이용하여 필요한 소프트웨어 하드웨어를 이용할 수 가 있다. 이러한컴퓨터 자원은 고객들이 소유 또는 관리 하기에는 너무 비싸지만 클라우드 컴퓨팅 서비스를 이용함으로써 비용절감의 효과가 있고 사업자들 입장에서는 고객들에게 사용 비용을 청구함으로써 고객과 사업자들 모두 상생할 수 있는 모델이다.

## 2. 클라우드 컴퓨팅 소개

미국 NIST(National Institute of Standard and Technology) 에서는 클라우드 컴퓨팅을 다음과 같이 정의한다: 최소한의 관리 및 서비스 제공자와의 연동으로 네트워크 상에 연결되어 있는 빠르게 제공되는 공유 컴퓨팅 자원들의 모델.

쉽게 말해서 인터넷 상에서 사용자나 회사에 소프트웨어 및 하드웨어를 제공하는 서비스를 말한다. 그럼 왜 갑자기 클라우드 컴퓨팅이 나타나기 시작했을까. 먼저 광대역 인터넷, 이동통신 및 위치기반 서비스등의 인터넷 기술 발전과 사용량 증가를 들 수 있다. 또한 그로인하여 증가한 너무나 많은 수평적 데이터를 이유로 들 수도 있다. 즉, 인터넷 과 웹이 등장하기 이전에는 데이터가 부족해서 문제였지만 지금은 거의모든 웹사용자들이 만들어내는 디지털 데이터로 인하여 관리 불가능한 정보의 홍수속에 노출되어 있다. 덧붙여서 컴퓨팅 자원들의 가격 인하및 수많은 오픈소스들의 등장으로

누구나 아이디어가 있으면 쉽게 서비스를 제공 판매 할 수 있는 기반이 형성되었다는 것이다.

앞에서 클라우드 컴퓨팅을 정의 하였지만 그 자체를 새로 나타난 무엇이라고 할 수는 없고 이미 우리가 사용하거나 들어본적이 있는 서비스를 통합해서 새로운 용어로 만들었다고 할 수도 있다. 즉 웹상에서 서비스를 제공하는 모델인 웹서비스, 웹으로 제공하는 이메일 서비스, 서버를 제공하는 호스팅 회사등 이미 우리에게 친숙한 서비스들을 모두 클라우드 컴퓨팅이라고 할 수 있다. 이러한 클라우드 컴퓨팅은 가상화 (virtualization)에 기반하여 이루어 질 수 있다. 가상화는 가상화 된 소프트웨어, 하드웨어, 네트워크, 컴퓨팅자원 등을 제공하면 사용자가 마치 집에서 컴퓨터를 쓰듯이 가상화된 소프트웨어등을 쓸수 있는 환경을 말하며 사용자의 사용량 사용시간등에 따라 가상환경 제공자는 요금 청구를 할 수 있다.

클라우드 컴퓨팅은 크게 세가지로 분류할 수 있다: Software As a Service (SaaS), Platform As a Service (PaaS), Infrastructure As a Service (IaaS). 또한 이 세가지 분류를 소프트웨어 기반인 SaaS 그리고 하드웨어 기반인 PaaS 와 IaaS 로 나눌수가 있다. 그 자세한 정의는 다음에서 나열한다.

## 2.1. 소프트웨어 기반

앞에서 분류하였듯이 소프트웨어에 기반한 클라우드 컴퓨팅은 SaaS 와 PaaS 로 나뉜다. SaaS 는 소프트웨어 즉 응용 프로그램을 제공해 주는 서비스이다. 주로 웹을 통하여 제공되는 응용프로그램 서비스로서 누구나 가입만 하면 프로그램을 따로 관리할 필요도 없고 때로는 다운로드 할 필요도 없이 이용할 수 있는 서비스이다. 미국에서 흔히 쓰는 Google Doc (<http://doc.google.com>) 이 그 예로서 구글이 제공하는 사무용 프로그램인 스프레드 시트, 워드, 폼, 슬라이드등을 그저 가입만 함으로써 이용할 수 있다. 또한 이메일을 제공하는 yahoo, google 그리고 동영상과 사진을 각각

공유할 수 있는 youtube, flickr 등이 있다. 또한 프로젝트 관리 프로그램을 제공하는 assembla.com 및 소셜네트워크를 제공하는 twitter, facebook 이 있다. 덧붙여서 기존의 세금 계산 프로그램을 웹에서 작동시키는 TurboTax online 이 있다.

## 2.2. 하드웨어 기반

PaaS 는 응용프로그램을 개발하거나 실행할 수 있는 플랫폼을 제공하는 서비스이다. 즉 응용 서버 관리, 데이터베이스 관리, 보안, 워크플로우관리 플랫폼을 제공하는 서비스로서 미국의 예로는 개발자가 개발된 프로그램을 구글 웹에 올려서 실행 테스트를 할 수 있는 구글 AppEngine 이나 개발된 Rails 프로그램을 올려 실행 및 퍼블리쉬 할 수 있는 EngineYard.com 을 그 예로 들 수 있다.

IaaS 는 거의 무한대인 가상 컴퓨팅 자원 인프라를 제공하는 서비스이다. 즉 사용자가 소유한 소프트웨어 및 프로그램을 실행할 수 있는 컴퓨팅 자원 및 저장 공간 즉 인프라 스트럭처를 제공하는 서비스이다. 미국에서의 예는 아마존 AWS(Amazon Web Services) 인 분산병렬컴퓨팅 인프라 EC2 와 가상저장공간 S3 그리고 마이크로소프트사의 Azure 가 있다.

사실 PaaS 와 IaaS 의 정의를 읽어보아 알겠지만 그 차이를 구분하기가 쉽지 않다. 개인적으로는 이 둘을 구분하는데 있어서 PaaS 는 웹 호스팅 서비스에 가깝고 IaaS 는 가상 서버 제공 서비스라고 생각한다.

## 2.3. 클라우드 컴퓨팅의 유의점

클라우드 컴퓨팅에서 가장 유의 할 점은 보안이다. 컴퓨팅 자원이 원격으로 존재하기 때문에 만약 서비스 제공자가 경쟁사 또는 적대국이라면 사용자의 귀중한 프로그램이나 정보가 노출되어 경쟁에서 생존할 수가 없다.

마찬가지로 서비스 제공자가 자원 관리에 실패하면 사용자의 프로그램이

제거될 수도 있다. 따라서 보안상에 대한 책임은 여전히 전적으로 사용자에게 달려있다. 따라서 서비스 제공자를 너무 신뢰하지 말고 항상 보안에 대한 주의를 기울여야 한다.

#### 2.4. 클라우드 컴퓨팅의 장점

이와같이 클라우드 컴퓨팅을 이용시 유의 할 점도 있지만 여러 이득이 있다. 먼저 사용자가 굳이 소프트웨어나 하드웨어를 구입하여 관리하지 않아도 서비스에 가입만 하면 쉽게 그 컴퓨팅자원을 이용 할 수 있는 장점이 있다. 더구나 아마존 AWS 에서 처럼 사용시간에 따른 비용처리를 할 수 있으므로 굳이 서비스에 가입할 필요도 없다. 그 밖에 원하는 자원을 더욱 확장할 수 있는 확장성 (scalability)이 좋다. 예를 들면 웹 방문자가 두 배로 늘었을때 가상 웹 응용 서버의 숫자를 두배로 늘리면 이론적으로는 가상서버가 늘어난 방문자들을 모두 책임질 수 있다. 또한 개발된 프로그램을 가상 서버에 올리는 것이 빠르고 용이하다. 그리고 서비스 제공자가 제공하는 API (Application Programming Interface)를 이용함으로써 서비스를 더욱 효과적으로 사용할 수 있다.

현재 미국에서 사용되는 클라우드 컴퓨팅 API 의 예는 아마존의 AWS API , 구글 Map API , 썬 마이크로 시스템 (현 오라클)의 Open Cloud API , VMWare 의 vCloud API , ServerPath 의 GoGrid API 등이 있다.

그럼 다른 관점중의 하나인 경제적효율성을 보도록하자. IT 관련이든 아니든 현대사회에서는 회사를 창립할 때 IT를 간과할 수 는 없다. 클라우드 컴퓨팅과 같은 사용시만 비용을 처리하는 모델은 창업자나 투자가 모두에게 비용절감이라는 이점을 제공한다. 즉 설비투자비용 (Capital expense)과 영업비용(Operating expense)의 차이이다. IT 자원을 직접 구매하는 설비투자비용보다 IT 자원을 빌려서 쓰는 영업비용이 훨씬 저렴하다. 따라서 클라우드 컴퓨팅은 Start-Up 회사를 차릴수 있는 보다 용이한 기회를

제공한다. 미국의 예를 들면 Start-Up 회사를 운영할 때 회사의 이메일은 구글의 이메일을 사용하고 프로젝트 관리는 assembla.com 에서 하며 회의는 skype 대화서비스를 이용하고 데이터베이스 및 웹 응용서버는 godaddy.com 이나 engineyard.com 을 이용한다.

### 3. 스토리지 가상화 플랫폼

#### 3.1. 스토리지 가상화 플랫폼

스토리지 서비스는 각 기업마다 굳이 그 구조를 밝히고 있지 않으므로 정확하게 그 구조를 설명하기는 쉽지 않다. 하지만 전형적인 데이터 베이스들을 클러스팅기반으로 연결하여 서비스를 제공하고 있으리라 짐작할 수 있다. 이러한 전형적인 relational 데이터 베이스들은 대부분의 사람들이 그 구조를 짐작 할 수 있으리라 믿음으로 이 보고서에서는 더이상 언급하지 않는다.

하지만 데이터용량이 폭주하는 요즈음은 전형적인 Relational 데이터 베이스의 복잡한 스키마(Schema) 구조나 Query 없이 간단하게 키와 값만으로 구성된 행 지향 데이터 베이스 (Column-oriented Database) - NoSQL 데이터 베이스라고도 한다 - 가 보다 유용하게 이용되고 있으므로 여기서 다루고자 한다.

예를 들어 Relational 데이터 베이스에 다음과 같은 표가 있다고 가정하자.

| EmpId | Lastname | Firstname | Salary |
|-------|----------|-----------|--------|
| 1     | Smith    | Joe       | 40000  |

|   |         |       |       |
|---|---------|-------|-------|
| 2 | Jones   | Mary  | 50000 |
| 3 | Johnson | Cathy | 44000 |

행 지향 데이터 베이스에서는 행 별로 데이터가 다음과 같이 저장된다.

1, 2, 3;

Smith, Jones, Johnson;

Joe, Mary, Cathy;

40000, 50000, 44000;

이러한 각 행 지향적인 데이터 저장은 데이터 웨어하우스와 같은 OLAP (Online Analytical Processing)에 적합하다.

또다른 예를 들어 Relational 데이터 베이스에 다음과 같이 데이터 밀도가 적은 표가 있다고 가정하자 - 보통 un-, 또는 semi-structured 데이터에 해당된다.

| EmpId | Lastname | Firstname | Salary |
|-------|----------|-----------|--------|
| 1     | Smith    |           | 40000  |
| 2     | Jones    | Mary      |        |

|   |  |       |       |
|---|--|-------|-------|
| 3 |  | Cathy | 44000 |
|---|--|-------|-------|

행 지향 데이터 베이스에서는 행 별로 데이터가 다음과 같이 저장된다.

1, 2, 3;

Smith, Jones;

Mary, Cathy;

40000, 44000;

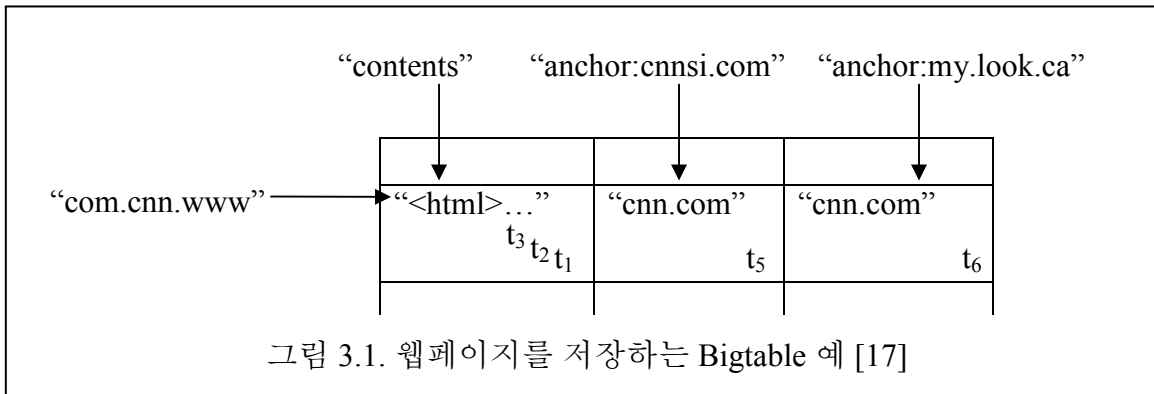
즉 Null 데이터들을 신경쓸 필요없이 행 별로 Null 이 아닌 데이터들만 저장함으로써 메모리 공간을 아낄 수 가 있다. 웹 다큐먼트와 같은 unstructured 데이터들을 저장한다고 생각할 때 이러한 적은 밀도의 데이터를 기존의 Relational 데이터베이스에 저장하여 저장 공간을 낭비하는 것 보다 행 지향 데이터 베이스에 저장하는것이 이득이다.

구글은 Bigtable 이라는 자체 GFS (Google File systems) 파일 시스템에서 실행되는 빠르고 성능 좋고 확장성 좋은 행 지향적인 데이터 베이스 시스템을 구현하여 웹 인덱싱, 구글 맵, 구글 리더 (reader), 구글 어스 (Earth), 유튜브, 구글메일등에 이용하고 있다. Bigtable 은 다차원적으로 정렬된 맵으로서 다음과 같이 행과 열의 키와 타임스탬프로써 값을 저장한다.

(row:string, column:string, time:int64) -> string 즉 썬

행 키는 family:qualifier 의 조합으로써 family 는 웹페이지의 anchor 등으로 정의 할 수 있고 qualifier 는 참조된 웹페이지 그리고 저장공간인 썬 (Cell)에는 링크 텍스트가 저장된다.

그림 3.1 은 웹페이지를 저장하는 Bigtable 의 예를 보여주고 있다 [17].



열의 이름 또는 키는 URL 주소를 역으로 표현한 것이다. Contents 행은 contents 를 family 로 하여 그 페이지의 내용물을 저장 하였고 anchor 행은 anchor 를 family 로 하여 그 anchor 가 표시하는 텍스트들을 저장한다. 특히 contents 행은 그 값들이 타임스탬프로 -  $t_1 < t_2 < t_3$  - 표현되어 있듯이 버전 별로 저장되어 있다.

좀 더 쉽게 Bigtable 을 다음과 같이 표현할 수 있다:

열 key 1 → {[행 key l, 값 l], [행 key m, 값 m], [행 key n, 값 n]}

열 key 2 → {[행 key i, 값 i]}, [행 key j, 값 j], [행 key k, 값 k]}

즉, 분산 되어 밀도가 적은 테이블을 저장 공간 낭비 없이 열과 행만으로 저장할 수 있다.

또 다른 예로 웹 페이지를 방문한 사용자 log 파일을 행지향 데이터베이스에 저장한다고 가정하여 그 스키마 모델을 구현해 보자. 로그 파일은 시간, 사용자 로그인 ID, 방문 url 등으로 저장된다고 가정한다.

그림 3.2 와 같이 열은 “시간” 으로 행 families 는 “사용자” 와 “http” 로 나누어서 각 쉘 값인 “사용자: login\_ID” 와 “http:URL” 값을 저장 할 수 있다.

| 열  | 행 families    |          |
|----|---------------|----------|
| 시간 | 사용자           | http     |
|    | 사용자: login ID | http:URL |

그림 3.2. 웹페이지 log 파일을 저장하는 Bigtable 스키마 구현 모델 예

### 3.2. 활용

스토래지 기반의 서비스를 제공하는 대표적인 미국 기업은 아마존이다. 특히 아마존 S3 (Simple Storage Service)는 사용자가 키를 지정한 후 객체들을 저장하고 언제 어디서나 객체들을 추출할 수 있는 구조이다. 또한 아마존은 다이나모 (Dynamo) 라는 행기반 DB 를 구축하고 있다.

아파치 HBase 프로젝트는 구글의 Bigtable 를 모델로 하여 하둡의 HDFS (Hadoop Distributed File Systems) 파일시스템 상에서 구현된 오픈 소스 Bigtable 이며 다음장에서 서술된 하둡 Map/Reduce 방식과 통합구현되어 있다 [20].

소셜 네트워킹 및 IT 세상을 서서히 뒤흔들고 있는 Facebook 은 자체 탐색 데이터를 효율적으로 처리하려는 시도로써 카산드라 (Cassandra) 행 지향 데이터 베이스를 성공적인 해법으로 내놓았고 구글과 아파치의 개방 프로젝트로 카산드라를 공개하였다. 이것은 아마존 다이나모의 분산구조와 구글의 Big Table 개념을 가져와 확장성과 분산성이 뛰어나다. 또 다른 소셜 네트워킹 사이트인 트위터 (Twitter)도 카산드라를 이용하고 있다.

### 3.3. 동향

많은 기업들이 대용량 데이터를 처리하기 위하여 no-SQL 또는 행 지향 데이터 베이스에 관심을 가지고 있고 또한 행 지향 데이터를 선택하는 중 이다. 하지만 아마존 다이나모를 제외한 MongoDB, CouchDB, 카산드라, HBase 등 여러 개방형 행 지향 데이터 베이스 중 어느것을 선택할 지에 대한 고민도 가지고 있다. 당연히 기업 특성 및 각 응용 프로그램에 따라 어떤 데이터 베이스를 선택할 지가 결정이 내려져야 한다. 하지만 카산드라나 HBase 가 분산성 및 확장성에 있어 장점이 있어 먼저 고려되는 듯이 보인다. HBase 는 다음장에 설명하는 Map/Reduce 계산을 이용하는 프로젝트에 적합하다. 그러나 하둡에 근간한 HBase 는 Name Space 노드가 멈추면 전체가 작동을 멈추지만 카산드라는 그럴 염려가 없다 [22].

#### 4. Map/Reduce 컴퓨팅 자원

웹 이전의 세계에서는 데이터가 부족해서 문제였으나 웹 이후 지난 16 여년간의 웹상에서의 데이터 축적은 데이터가 너무 많아서 문제이다. 왜냐하면 데이터 분석을 통하여 필요한 정보를 추출하는 시간과 비용이 너무 비싸지기 때문이다. 더구나 수많은 아날로그 정보를 디지털화 함으로써 축적된 방대한 데이터도 문제이고 아날로그 정보를 디지털화 하는 단순하지만 비싼 응용 시스템도 문제이다. 클라우드 컴퓨팅 특히 그 Map/Reduce 방식은 이러한 방대한 데이터 중 단순하지만 비싼 데이터 추출 알고리즘을 병렬 계산 방식으로 쉽게 구현하는 해법을 제공한다.

클라우드 컴퓨팅의 활용방안 및 기대효과는 다음과 같은 1) 구글에서 왜 Map/Reduce 방식이 고안되었는지와 2) 뉴욕타임즈에서의 예를 들어 설명하고자 한다. 덧붙여서 3) 아파치 하둡 프로젝트를 소개한다.

구글 연구원들이 당면했던 문제 중의 한 예는 수억의 사람들이 방문한 사이트들의 사용자 로그 정보에서 어떻게 하면 쉽고 빠르고 싸게 가장 방문빈도가 높은 사이트를 찾아 내어 방문빈도를 정렬하는가 하는 것이었다.

즉, 천문학적으로 방대한 데이터에서 비교적 쉬운 계산 알고리즘을 돌려서 빠르게 해법을 찾아내는 시스템의 구축이었다. 그래서 병렬 프로그래밍에서의 전통적인 MapReduce 알고리즘을 해법으로하는 시스템을 구글내에서 구축하게 된 것이다.

또한, 그 이후 미국에서는 전자상거래 사이트인 아마존에서 AWS 라는 아마존 클라우드 컴퓨팅 서비스를 구축하여 누구든지 싼 비용으로 클라우드 컴퓨팅을 이용하게 하였다. 이서비스를 이용하면 비교적 단순한 프로그래밍 로직이지만 방대한 데이터를 가진 프로그래밍을 Map/Reduce 방식으로 돌려서 원하고자 하는 답을 찾아낼수 있다. 예를 들면, 2007 년도에 뉴욕타임즈에서 1851 년도 부터의 약 일천일백만개 되는 기사(약 4 테라 바이트 tiff 파일)을 pdf 파일 방식으로 바꿀려고 하였다. Tiff 파일을 pdf 파일로 바꾸는 아주 간단한 프로그래밍이지만 그 데이터 양이 천문학적으로 방대하여 그 비용과 시간이 너무 비쌌었다. 하지만 데렉이라는 소프트웨어 엔지니어가 아마존 AWS 의 EC2 와 S3 를 이용하여 단 \$240 로 24 시간내에 모두 pdf 파일로 바꾸어 냈다.

또한 아파치 하둡 (<http://hadoop.apache.org/>) 프로젝트는 이러한 Map/Reduce 병렬 계산 시스템을 구축할 수 있는 소프트웨어 플랫폼을 무료로 제공한다. 따라서 여러대의 컴퓨터 (하드웨어) 만 있으면 누구나 아마존의 AWS EC2 와 같은 Map/Reduce 시스템을 구축할 수 있다. 대한민국 정부, 연구소나 학교에서 Map/Reduce 시스템을 구축하는 프로젝트를 시발하고 Hadoop 을 이용한다면 Map/Reduce 소프트웨어를 처음부터 구현할 필요없이 Hadoop 을 그 시스템에 최적화 하는 일만 함으로써 대한민국 기업, 연구소, 기업체 및 일반인등을 위한 Map/Reduce 클라우드 컴퓨팅 시스템을 구현 할 수 있을것 이다.

본 기술 동향 조사서의 최종본에는 앞서 서술한 예들을 구체적으로 서열하고 미국에서의 클라우드 컴퓨팅 서비스 업체들 특히 Map/Reduce 서비스를

제공하는 업체들을 조사하고 그 특징을 서술한다. 그리고 대한민국내에서 그러한 서비스를 활용할 수 있는지 활용방안도 제시한다. 또한 아파치 하둡에 대한 조사를 통하여 대한민국 내에서 하둡서비스를 설치가능한지에 대한 활용방안도 조사하고 서술한다.

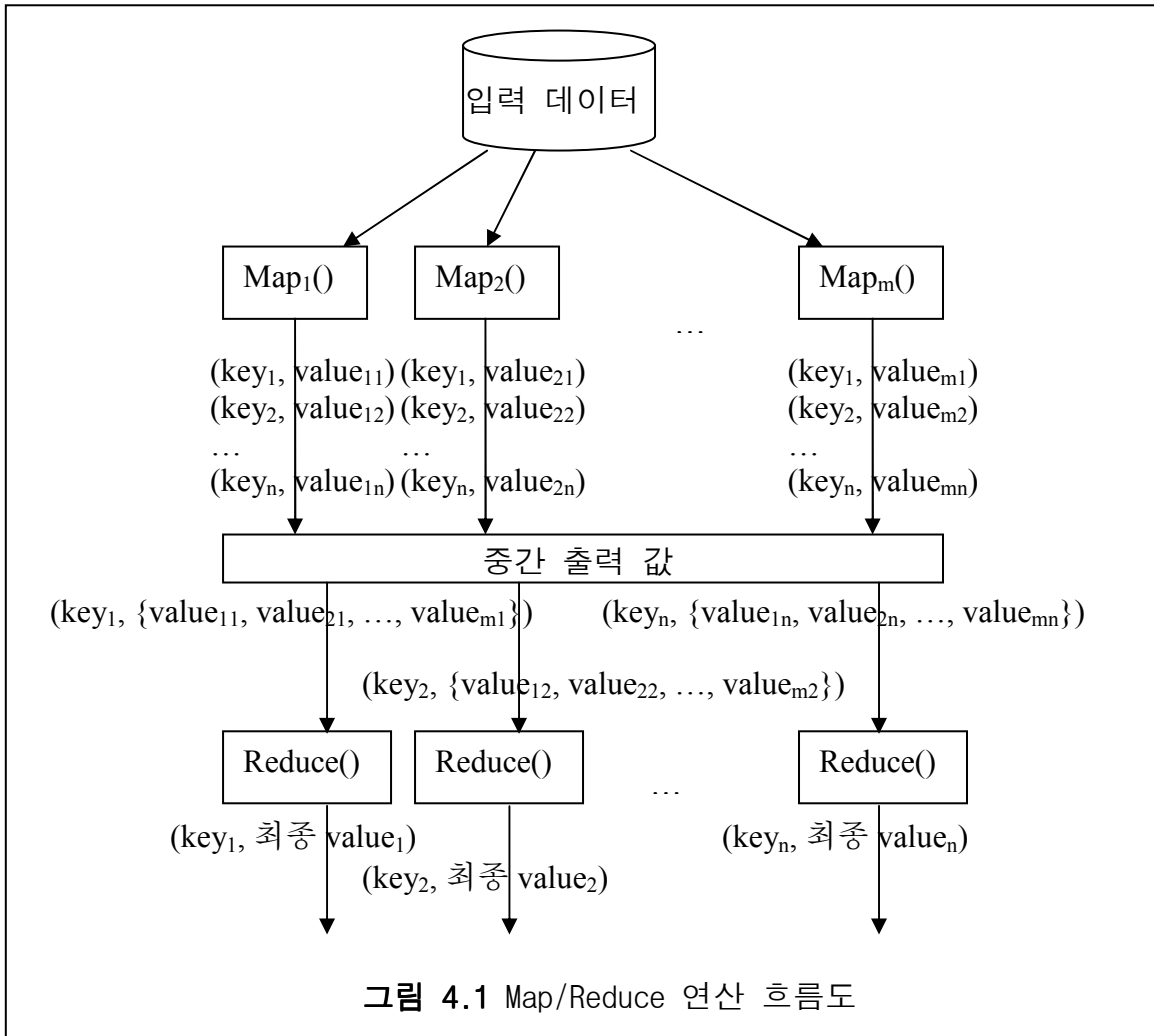
그리고, 미국내에서의 위와 같은 Map/Reduce 시스템을 이용한 응용 예들을 분석하여구체적으로 서술하고 대한민국에서의 활용방안에 대한 예제를 제시하며 다음과 같은 기대효과를 좀더 자세히 나열한다. 즉, Map/Reduce 클라우드 컴퓨팅 시스템을 통하여 방대한 데이터를 쉽고 빠르게 처리할 수 있는 기반이 구현되면 지질 및 천문기상 데이터를 분석하는데 이용 가능하며 웹 상에서는 사용자들의 웹 사용 행동 방식을 분석함으로써 사용자들 구미에 맞는 정보를 제공가능하다. 또한, 미국방부에서 연구는 것 처럼 방대한 웹사이트 데이터를 분석하여 적대 조직을 발견하거나 테러를 방지할 수 있는 시스템을 구현하는데 이용 할 수 있을 것이다.

#### 4.1. 정의

Map/Reduce 컴퓨팅은 Map/Reduce 방식만을 이용하는 제한적 병렬 컴퓨팅으로 정의 할 수 있다. 따라서 Map/Reduce 컴퓨팅 환경에서는 개발자가 어떤 데이터계산을 위하여 Map()과 Reduce() 기능만 개발하면 나머지 병렬 계산 기능인 병렬화, 데이터 분산, load balancing, fault tolerance 등은 컴퓨팅 환경이 제공해 준다. Map/Reduce 방식은 Functional 프로그래밍 언어인 LISP에서 주로 쓰던 기능으로써 간단하지만 방대한 (Tera- 및 Peta-바이트) 데이터를 처리해야만 하는 계산에 주로 쓰인다. 그 예는 앞에서 언급한 뉴요타임즈 및 웹 로그 파일의 경우이다.

Map()은 데이터를 (key, value)의 쌍으로 변환하는 함수이다. Map() 함수를 병렬계산기의 각 프로세서에서 실행하면 각 프로세서에 분산해서 입력된 데이터들이(key, value)의 쌍으로 변환된 중간 데이터가 생성된다. 그림 1

에서 처럼 입력데이터가 각 Map() 함수에 분산되어 보내지면 각 Map<sub>1</sub>() 함수는 주어진 입력데이터를 (key<sub>1</sub>, value<sub>11</sub>), (key<sub>2</sub>, value<sub>12</sub>), ..., (key<sub>n</sub>, value<sub>1n</sub>)의 중간 출력값으로 변환시킨다. 다른 Map() 함수들도 병렬연산으로 동시에 각 출력값을 계산한다.



Reduce()는 이 변환된 중간 데이터들을 모아서 다시 같은 키를 가진 출력데이터로 만드는 함수로서 같은 키를 가진 데이터들이 각 프로세서에서 최종 value 를 계산한다. 즉 키 별로 병렬로 실행된다. 즉 그림 4.1 에서 처럼 key<sub>1</sub> 은 (key<sub>1</sub>, {value<sub>11</sub>, value<sub>21</sub>, ..., value<sub>m1</sub>})으로 값들을 모아서 (key<sub>1</sub>, 최종 value<sub>1</sub>)의 최종값을 계산한다. 다른 Reduce() 함수도 병렬연산으로 각

key 값을 계산한다. 여기서 최종  $value_1 = function(value_{11}, value_{21}, \dots, value_{m1})$  이고  $function()$ 은 주어진 입력과 해법에 따라 개발자가 개발한다.

이 Map/Reduce 연산 방식은 Map()이 계산되기 전까지 Reduce()는 실행이 되지 않으므로 Map() 계산이 병목이 된다.

예를 한번 들어보자. 갭이라는 회사에서는 회사가 운영하는 웹사이트에서 어느 웹 페이지에 사용자들이 가장 많이 방문하였는지를 알아내어 웹 페이지들을 개정하려고 한다. 마침 그 웹 서버에서는 그림 4.3 과 같이 웹 방문기록이 남아있는 웹 로그 파일 (log\_visit.txt)을 항상 자동적으로 생성해 왔다. 개발자는 이 로그파일을 읽어서 방문한 url 을 셀수 있는 간단한 Map()함수를 개발한다. 즉 Map() 함수는 log\_visit.txt 파일을 읽어서 (key, value)의 쌍이 (url, 방문횟수)로 구현되는 출력을 만들어 낸다.

예를들면 ( "http://www.gap.com/index.html" , 321)의 쌍으로 출력이 생성된다. 이때 키 값은 "http://www.gap.com/index.html" 이고 방문회수 321 이 그 값이 된다. 이 로그 파일의 크기가 작으면 문제가 없지만 그 크기가 peta-바이트이면 그림 4.2 와 같은 Map/Reduce 환경에서 Map()함수는 로그파일을 나누어서 입력시킨 분산된 Map() 병렬연산기에서 실행이 되어 각 프로세서 마다 중간 출력 데이터를 얻는다. 즉 Map<sub>1</sub>()

프로세서에서는( "http://www.gap.com/index.html" , 321), Map<sub>2</sub>()

프로세서에서는( "http://www.gap.com/index.html" , 4531) 같은 값을 얻는다.

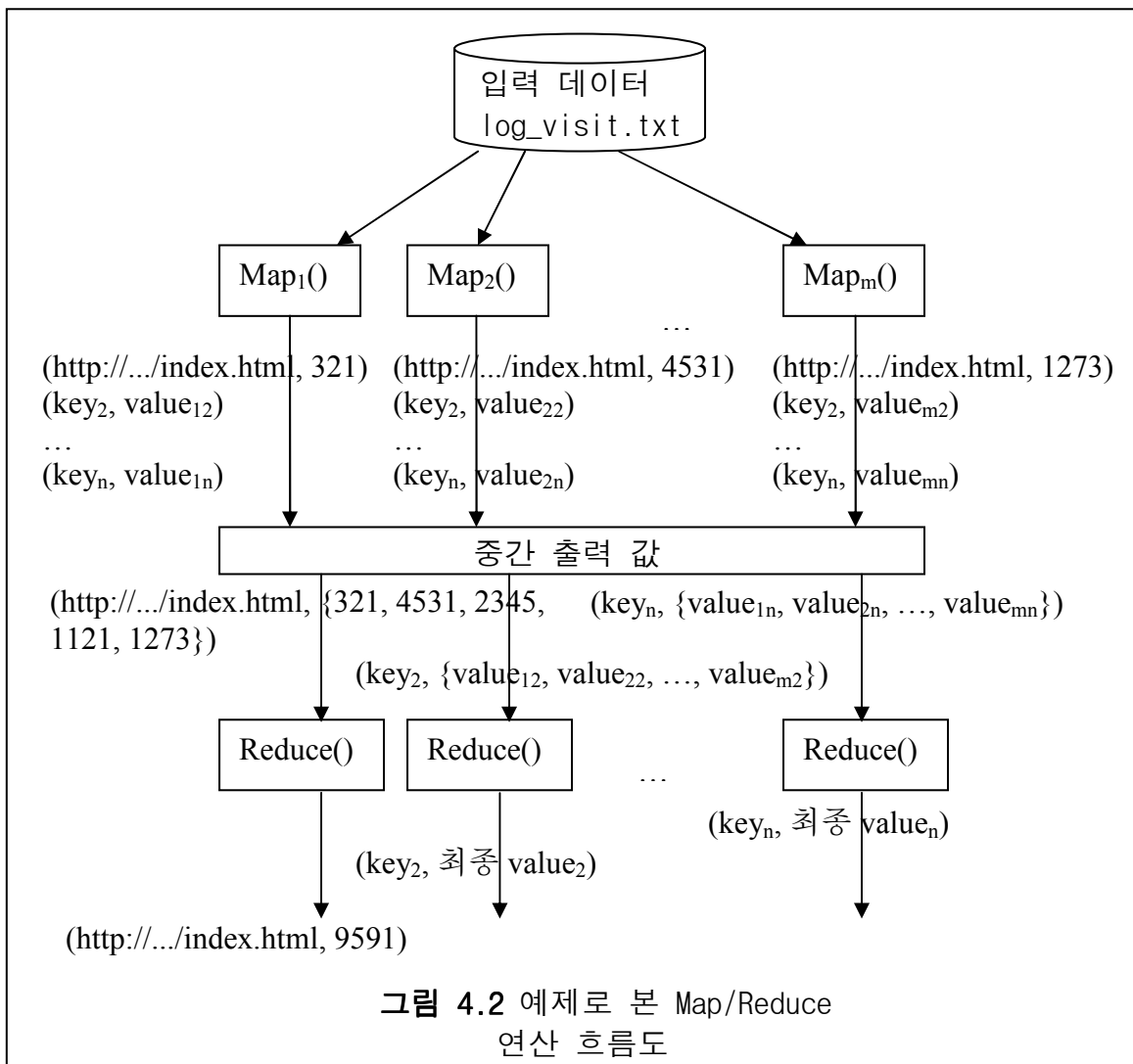
모든 Map()계산이 끝나면 다른 병렬 연산인 Reduce() 함수에 키 별로 데이터가 분산되어서 총 방문횟수가 병렬로 계산이 되어 그림 4.2, 4.4 와 같은 (url, 총방문횟수) 쌍의 출력값이 생성된다. 즉, 그림 4.2 에서 처럼 5 개의 프로세서들이 있다고

가정하였을때 "http://www.gap.com/index.html" 를 Map() 함수로 계산하여 각

프로세서 별로 {321, 4531, 2345, 1121, 1273} 의 방문횟수를 가진  
 중간출력값을 생성한다. 그러면 reduce() 함수에서 이 중간출력값을  
 입력으로 하여 키 별로 각 프로세서에서 총 방문횟수를 계산하여

```
( "http://www.gap.com/index.html" , {321, 4531, 2345, 1121, 1273})=>
( "http://www.gap.com/index.html" , sum(321, 4531, 2345, 1121, 1273))=>
( "http://www.gap.com/index.html" , (321 + 4531 + 2345 + 1121 + 1273))
=> ( "http://www.gap.com/index.html" , 9591)
```

라는 최종 출력값을 얻는다.



이 예제에서의 Reduce() 함수는 sum()으로써 각 웹사이트 페이지를 방문한 중간값을 합산하는 함수이다. 따라서, 연산하고자 하는 프로그램의 종류에 따라 개발자가 거기에 맞는 Reduce() 함수를 개발 해야 한다.

|  |   |                               |      |                                  |      |                                  |      |                              |       |
|--|---|-------------------------------|------|----------------------------------|------|----------------------------------|------|------------------------------|-------|
| <pre> http://www.gap.com/index.html http://www.gap.com/contacts.html http://www.gap.com/products.html http://www.gap.com/index.html http://www.gap.com/help.html http://www.gap.com/index.html http://www.gap.com/contacts.html http://www.gap.com/products.html http://www.gap.com/contacts.html http://www.gap.com/products.html http://www.gap.com/index.html http://www.gap.com/help.html ... </pre> | <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="padding: 5px;">http://www.gap.com/index.html</td> <td style="padding: 5px; text-align: right;">9591</td> </tr> <tr> <td style="padding: 5px;">http://www.gap.com/contacts.html</td> <td style="padding: 5px; text-align: right;">5678</td> </tr> <tr> <td style="padding: 5px;">http://www.gap.com/products.html</td> <td style="padding: 5px; text-align: right;">6782</td> </tr> <tr> <td style="padding: 5px;">http://www.gap.com/help.html</td> <td style="padding: 5px; text-align: right;">23456</td> </tr> </table> | http://www.gap.com/index.html | 9591 | http://www.gap.com/contacts.html | 5678 | http://www.gap.com/products.html | 6782 | http://www.gap.com/help.html | 23456 |
| http://www.gap.com/index.html  | 9591  |                               |      |                                  |      |                                  |      |                              |       |
| http://www.gap.com/contacts.html   | 5678  |                               |      |                                  |      |                                  |      |                              |       |
| http://www.gap.com/products.html   | 6782  |                               |      |                                  |      |                                  |      |                              |       |
| http://www.gap.com/help.html   | 23456   |                               |      |                                  |      |                                  |      |                              |       |
| <b>그림 4.3. 웹 방문 로그 파일</b>  | <b>그림 4.4. 웹 방문 출력 파일</b>   |                               |      |                                  |      |                                  |      |                              |       |

## 4.2. 아파치 하둡과 피그

아파치 하둡 (Apache Hadoop)은 본래 웹 서치 엔진인 아파치 너치 (Nutch) 프로젝트의 기본으로 구현되었다 [7, 8]. 하둡은 Map/Reduce 연산과 하둡 분산 파일 시스템 (HDFS: Hadoop Distributed File Systems)을 구현한 오픈 프로젝트 프레임워크이다 [7]. 하둡의 가장 기본적인 아이디어는

“연산기능을 데이터에 근접시키자” 이다. HDFS는 구글의 구글 파일 시스템 (GFS)에서 영감을 얻었으며 [9] 각 연산 기능을 가능한 한 연산 데이터에 근접하도록 하여 대용량 데이터 전송을 줄이며 연산 효율을 높이도록 구현되어 있다. 따라서, 네트워크상에 연결된 수천개의 컴퓨터에 하둡을 설치하여 클러스터 환경을 구축하면 하둡의 HDFS는 거대한 파일을 신뢰성 있게 저장할 수 있다. 그리고, 이러한 하둡의 클러스터 환경에서 Map/Reduce 병렬연산이 가능하다. 또한, 한 노드가 작동하지 않아도 나머지 노드들은

여전히 작동할 수 있도록 하둡 프레임워크이 자동으로 처리할 수 있는 높은 신뢰성을 보유한다.

하둡을 설치하여 자바 이클립스 (eclipse) IDE 상에서 예제를 직접 돌리고 싶으면 필자의 블로그를[12] 방문하여 예제를 따라해 보기 바란다 . 만약 의문사항이 있으면 블로그에 방문 일지를 남기면 즉시 답해주도록 노력하겠다.

아파치 피그 (Pig)는 하둡 프레임 상에서 작동하는 피그 프레임과 피그 라틴 (Pig Latin) 언어로 구현되어 있다 [10]. Map/Reduce 코드를 작성하는 것은 비교적 길고 복잡하다. 하지만 피그 라틴 코드로 같은 기능을 구현하는 것은 상대적으로 짧고 간단함으로써 피그 라틴 코드를 이용하여 Map/Reduce 코드를 작성 하는 것이 편리하다. 야후 같은 경우는 채용 인원의 60% 정도가 피그 이용자라고 한다. 피그 라틴 코드는 피그 프레임 상에서 하둡 코드로 변환이 되어 하둡 환경에서 실행된다.

#### **4.3. 산업체의 활용 동향**

현재 아마존, AOL, ebay, Facebook, 구글, Hulu, IBM, Twitter, Yahoo 등 많은 기구들이 적게는 4개 많게는 4000개 이상의 노드를 클러스트하여 하둡을 이용하고 있다 [13]. 그리고 대한민국에서도 하둡 사용자 그룹이 이에 관한 사이트를 유지하고 있다 [14].

아마존 Elastic Map/Reduce 는 아마존 웹서비스 (AWS)로서 EC2 (Amazon Elastic Compute Cloud) 와 S3 (Amazon Simple Storage Service)의 웹기반에서 작동하는 하둡 프레임을 이용하고 있다. 아마존 Elastic Map/Reduce 를 이용하면 하둡 환경을 설치 하거나 유지하는등의 부수적인 일에 시간을 뺏길 필요 없이 웹 인덱싱, 데이터 마이닝, 로그 파일 분석, 데이터 웨어 하우스, 재정 분석, 머신 러닝, 과학 시뮬레이션, 바이오 정보학과 같은 대용량 데이터 분석 및 처리에 집중 할 수 있다.

Cloudera 는 기업등을 위한 보다 확장된 사용성과 관리성 기능을 가진 하둡의 상용화된 버전인 CDH (Cloudera' s Distribution for Hadoop)를 구현하였다 [15]. IBM 도 InfoSphere Biginsights 라는 하둡에 기반한 클라우드 컴퓨팅을 구현하였다.

AOL 은 사용자 행동 분석에, Facebook 은 로그 저장과 분석, 머신러닝에 Hulu 는 로그 저장과 분석에 하둡을 이용한다. Ebay 는 서치 최적화 및 연구에 LinkedIn 은 친구 찾기기능에 하둡과 피그를 이용한다.

Twitter 는 Cloudera 의 CDH2 하둡을 이용하여 트위터 데이터 및 로그파일 저장과 분석에 피그를 이용한 Map/Reduce 연산을 한다.

야후는 4000 개의 노드를 가진 하둡 클러스터를 가지고 광고 시스템 및 웹서치에 관한 연구를 지원한다. 야후 채용의 60%이상은 피그 개발자들이다.

## 5. 결론

웹의 등장과 IT 산업의 발전 그리고 네트워크 기반 및 데이터 전송 속도의 가속화 등으로 인하여 인류는 수많은 컴퓨팅 및 스토리지 자원등을 공유할 수 있는 기반이 형성되었고 그동안 접하지 못했던 대용량의 데이터를 가지게 되었다. 이러한 컴퓨팅 및 스토리지 자원등을 가상화에 기반을 두고 사용자에게 제공하여 사용시간 및 사용량 등에 따라 요금을 부과 하는 클라우드 컴퓨팅 개념이 보편화 되고 있다. 또한 인류는 데이터양이 너무 많음으로써 데이터 검색, 데이터 통합, 데이터 공유 및 데이터 분석등이라는 새로운 과제에 당면한다. 이러한 대용량 데이터 처리 문제를 해결하기 위하여 병렬 처리 시스템의 한 해결 방법인 Map/Reduce 계산방식인 하둡등이 각광을 받기 시작하였고 또한 이러한 대용량 데이터를 저장하고 빠르게 처리 할 수

있는 no-SQL - 또는 행 지향 - 데이터 베이스인 HBase, Cassandra, CouchDB, MongoDB 등이 각광을 받기 시작하였다.

미국과 같이 소프트웨어가 발전한 곳에서는 여러 IT 기업들이 이러한 기술들을 2008 여년 부터 서서히 보편적으로 사용하기 시작하였다.

대한민국과 같이 하드웨어가 발전한 곳은 이러한 클라우드 컴퓨팅 기반은 이미 확보되어 있으나 하둡이나 no-SQL 데이터 베이스같은 소프트웨어 기반 기술을 공유하거나 토의 하는 모습은 드문듯 하다. 정부나 학계 산업계등에서 협력을 하면 이러한 대용량 데이터 관련 연구등에 선도적으로 참여 할 수 있으리라 본다.

## 6. 참고문헌

- [1]. “Introduction to Cloud Computing” , Jongwook Woo, the 10<sup>th</sup> KOCSEA (Korean Computer Scientists and Engineers Association in America) Technical Symposium, Las Vegas (Dec 18-19, 2009)
- [2]. “Introduction to Cloud Computing - for Startups and Developers” , Lew Tucker, Ph.D. CTO, Cloud Computing, Sun Microsystems, Inc.
- [3]. “Introduction to Cloud Computing - for Enterprise Users” , Lew Tucker, Ph.D. CTO, Cloud Computing, Sun Microsystems, Inc.
- [4]. “Google’ s Parallel Programming Model and Implementation Map/Reduce ” , Klara Nahrstedt and Sam King, UIUC
- [5]. “How to painlessly process terabytes of data” , John R. Gilbert, UCSB
- [6]. “Map/Reduce Theory and Implementation” , Christophe Bisciglia, Aaron Kimball, & Sierra Michels-Slettvet, University of Washington and Google

- [7]. Apache Hadoop Project, <http://hadoop.apache.org/>
- [8]. Apache Nutch Project, <http://nutch.apache.org/>
- [9]. “The Google File System” , Sanjay Ghemawat, Howard Gobioff, and Shun-Tak Leung, 19th ACM Symposium on Operating Systems Principles, Lake George, NY, October, 2003
- [10]. Apache Pig Project, <http://pig.apache.org/>
- [11]. Apache Hive Project, <http://hive.apache.org/>
- [12]. “Hadoop Example” , Jongwook Woo, <http://dal-cloudcomputing.blogspot.com/2009/08/hadoop-example-mymaxtemperaturewithcomb.html>
- [13]. Who uses Hadoop?, <http://wiki.apache.org/hadoop/PoweredBy>
- [14]. Hadoop Korean User Group, <http://www.hadoop.co.kr>
- [15]. Cloudera’ s distribution for Hadoop, <http://www.cloudera.com/hadoop/>
- [16]. IBM InforSphere BigInsights, <http://www-01.ibm.com/software/data/infosphere/hadoop/>
- [17]. Fay Chang, Jeffrey Dean, Sanjay Ghemawat, Wilson C. Hsieh, Deborah A. Wallach, Mike Burrows, Tushar Chandra, Andrew Fikes, Robert E. Gruber , “Bigtable: A Distributed Storage System for Structured Data” , OSDI '06: Seventh Symposium on Operating System Design and Implementation, Seattle, WA, November, 2006.
- [18]. MongoDB, <http://www.mongodb.org>
- [19]. Apache CouchDB, <http://couchdb.apache.org/>
- [20]. Apache HBase, <http://hbase.apache.org/>
- [21]. Apache Cassandra, <http://cassandra.apache.org/>
- [22]. Adku , HBase vs Cassandra, <http://blog.adku.com/2011/02/hbase-vs-cassandra.html>

[23]. Apache Cassandra Wiki,

<http://wiki.apache.org/cassandra/ArticlesAndPresentations>