논문 2012-49CI-1-2

# Hadoop 클러스터를 위한 모니터의 설계 및 구현

## ( Design and Implementation of a Monitor for Hadoop Cluster )

금 태 훈*, 이 원 주**, 전 창 호***

( Tae Hoon Keum, Won Joo Lee, and Chang Ho Jeon )

요 약

본 논문에서는 Hadoop 클러스터의 노드 정보와 작업 정보를 실시간으로 수집할 수 있는 새로운 모니터를 제안한다. 이 모니터는 Hadoop클러스터의 노드 정보와 작업 정보를 수집하는 Agent, 수집된 정보를 분석하고 데이터베이스에 저장하는 Collector로 구성된다. 또한 Collector를 Hadoop 클러스터에 참여하지 않은 새로운 노드에 위치시킴으로써 분석과정에서 발생하는 오버헤드로 인한 Hadoop의 작업지연을 제거한다. 제안한 모니터를 구현하고 실험적 클러스터에 적용함으로써, dead 노드의 발생을 실시간으로 파악할 수 있었다. 또한, Hadoop의 작업수행 과정에서 비효율적인 과정을 발견하고 개선함으로써 작업수행 시간을 단축시킬 수 있었다.

Abstract

In this paper, we propose a new monitor for collecting job information from Hadoop clusters in real time. This monitor is made of two programs called Collector and Agent. Agent collects Hadoop cluster's node information and job information, and Collector analyzes the collected information and saves it in a database. Also, Collector was placed in a new node outside the Hadoop cluster so that it does not affect Hadoop's work and will not cause overload. When the proposed monitor was implemented and applied, the testbed cluster was able to detect the occurrence of dead nodes immediately. In addition, we were able to find Hadoop jobs which were inefficient and when we modified such jobs to further enhance the performance of Hadoop.

**Keywords** : 클라우드 컴퓨팅(Cloud computing), 클러스터 모니터링(Cluster monitoring), Hadoop

## Ⅰ. 서 론

최근 클라우드 컴퓨팅[1]에 대한 관심이 높아지면서 클라우드 컴퓨팅을 위한 플랫폼[2~3]이 등장하고 있다. 특히, Hadoop[2]은 Apache의 오픈소스 기반의 플랫폼으로써 업데이트가 활발히 이루어지고 있으며, Apache의 검색엔진 Nutch의 분산처리 모델로 사용되고 있다. 또한, Amazon에서는 Hadoop을 이용하여 EC2[4] 서비스를 하고 있다. 이처럼 Hadoop은 클라우드 컴퓨팅을 위한 플랫폼으로 각광 받고 있다.

Hadoop은 여러 노드로 클러스터를 구성하고, 그 클러스터를 하나의 자원으로 사용할 수 있도록 HDFS (Hadoop Distributed File System)와 MapReduce 분산처리 모델을 제공한다. 이러한 Hadoop 클러스터를 효율적으로 운영하기 위해 노드들의 상태와 Hadoop 작업 정보를 수집하여 관리자에게 제공하는 모니터에 대한 연구가 진행되고 있다.[5~6] 하지만, 기존의 모니터는 수집한 데이터를 분석하기 위한 지연시간이 발생하여 실시간으로 모니터링을 할 수 없거나, 노드들의 상태정보

* 정회원, LG전자 MC 사업본부
(LG Electronics, Mobile Communications Company)
** 정회원, 인하공업전문대학 컴퓨터정보공학부 컴퓨터정보과
(Department of Computer Science, Inha Technical College)
*** 평생회원, 한양대학교 ERICA 캠퍼스 컴퓨터공학과
(Department of Computer Science & Engineering, Hanyang University ERICA Campus)
접수일자: 2011년8월9일, 수정완료일: 2011년12월30일

를 수집할 수 없는 문제점이 있다.

이러한 문제점을 해결하기 위해 본 논문에서는 Hadoop 클러스터의 노드 정보와 작업 정보를 실시간으로 수집할 수 있는 모니터를 제안하였다. 이 모니터는 Hadoop클러스터의 노드 정보와 작업정보를 수집을 하는 Agent와 수집된 정보를 분석하고 데이터베이스에 저장하는 Collector로 구성된다. 또한 Collector를 Hadoop 클러스터에 포함되지 않는 새로운 노드에 위치시킴으로써 분석과정에서 발생하는 오버헤드가 Hadoop의 작업에 영향을 미치지 않도록 한다.

본 논문의 구성은 다음과 같다. II장에서 클라우드 컴퓨팅과 Hadoop, 그리고 클러스터를 위한 모니터에 대하여 설명한다. III장에서는 제안한 모니터의 설계 및 구현과 동작에 대해 기술한다. IV장에서는 성능평가 및 분석에 대해 자세히 기술한다. 그리고 V장에서 결론을 맺는다.

## Ⅱ. 관련 연구

Hadoop 클러스터를 위한 모니터로는 Chukwa와 Mochi가 있다. Chukwa와 Mochi에 대하여 설명한다.

### 1. Chukwa

Chukwa는 기존의 모니터에서 제공하는 모든 노드의 상태 정보뿐만 아니라, HDFS 및 MapReduce의 정보까지 모니터링 한다. Chukwa의 동작 방식은 각 노드에 설치된 Agent가 Hadoop의 로그파일을 수집하여 Collector로 전송을 한다. 그리고 Collector는 수집한 정보를 하나의 아카이브파일로 HDFS에 저장하고, 이를 MapRuduce[7]를 통해 분석 후, 그 결과를 데이터베이스에 저장한다. 사용자는 웹으로 구성된 HICC(Hadoop Infrastructure Care Center)를 통해 정보를 확인할 수 있다. 이러한 Chukwa의 구조 및 동작은 그림 1과 같다.

그림 1 Chukwa의 구조는 대용량 Hadoop 클러스터에서 발생한 로그 파일을 분석하는데 효율적이다. 하지만 데이터 분석 주기가 기본적으로 5분마다 수행하도록 설정되어 있으며, 분석과정에서 Hadoop의 MapReduce를 사용함으로써 사용자에게 실시간으로 정보를 제공하지 못한다. 즉, 수집한 정보를 HDFS에 저장하고, 다시 MapReduce에서 로드하여 작업을 분산시키는데 따른 오버헤드가 발생한다. 따라서 사용자가 임의로 분석주기를 짧게 변경하여도 MapReduce의 작업지연 때문에

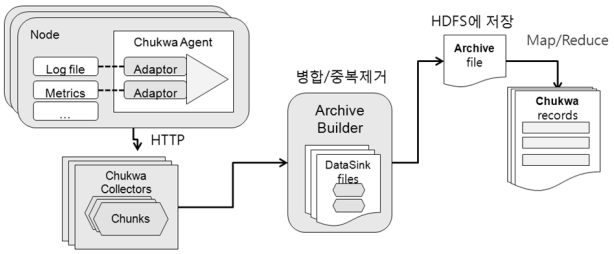


그림 1. Chukwa의 구조 및 동작
Fig. 1. Chukwa's architecture and behavior.

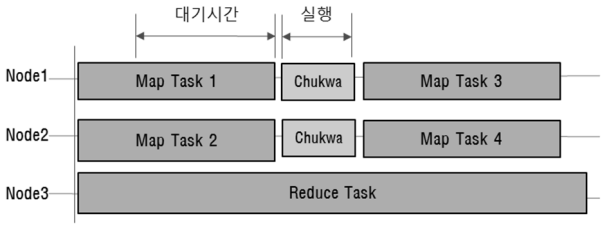


그림 2. Chukwa 작업이 대기할 경우
Fig. 2. Chukwa task's waiting time.

실시간으로 정보를 볼 수 없다.

Chukwa의 제안자가 2000개의 노드로 구성된 Yahoo 클러스터에서 실험한 결과, 약 3분의 분석 시간이 소요되었으며, 본 논문의 실험에서 사용된 10개의 노드로 구성된 클러스터에서는 20초의 분석 시간이 소요되었다.

또한, 그림 2와 같이 현재 MapReduce 수행중인 작업이 클러스터 전체를 사용하고 있을 경우, 자원을 선점하지 못하고 대기하는 상황이 발생한다. 이러한 문제 때문에 Hadoop 클러스터를 위한 모니터는 Hadoop과 결합도를 낮추어 설계해야 한다.

### 2. Mochi

Mochi는 Chukwa와 동일하게 Hadoop의 로그 파일을

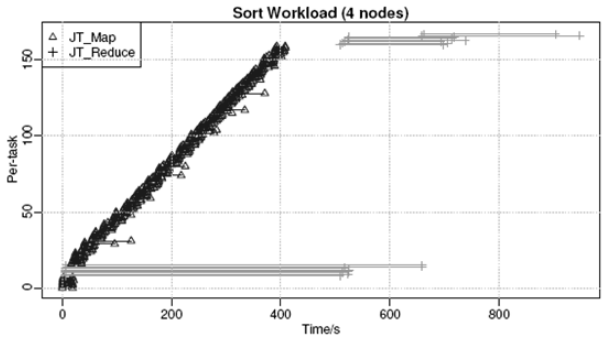


그림 3 Mochi의 MapReduce 작업정보 가시화
Fig. 3. Mochi's visualization of MapReduce job information.

수집하여 분석한 후, 사용자에게 보여주는 모니터이다. Mochi는 Hadoop의 설정을 변경하지 않고 로그 파일을 수집한다. 또한 수집한 정보를 분석하기 위해 MapReduce를 사용하지 않고, 별도의 Parser를 사용한다. Mochi는 모니터의 구조나 정보 수집 방식보다 MapReduce 작업 정보를 가시화하는데 중점을 두었다. Mochi에서 MapReduce 작업 정보를 가시화 하면 그림 3과 같다.

Mochi는 MapReduce의 작업정보 수집 및 가시화에 중점을 두었기 때문에 노드의 상태 정보는 수집하지 않는다. 따라서 노드의 상태 정보와 MapReduce 작업 정보를 모두 수집하는 새로운 모니터가 요구된다.

따라서 본 논문에서는 Chukwa와 Mochi의 문제점을 고려한 새로운 모니터를 제안한다.

## Ⅲ. Remon-HC 모니터

본 논문에서는 Hadoop 클러스터를 실시간으로 모니터 할 수 있는 RTM(Real-time monitoring for Hadoop cluster) 모니터를 제안한다. Remon_HC 모니터는 Chukwa, Mochi와 동일하게 Hadoop 로그 파일을 수집하고 분석한다. 그리고 분석 결과를 데이터베이스에 저장하고, 사용자 인터페이스를 통해 사용자에게 정보를 제공한다.

### 1. Remon-HC 설계

Remon-HC 모니터의 구조는 실시간으로 정보를 사용자에게 보여주기 위해 Collector를 Hadoop 클러스터 내에 설치하지 않고, 수집된 정보의 분석을 위해 MapReduce를 사용하지 않는다. 즉, 별도의 모니터링 서버에서 Collector를 구동시키고, 분석까지 완료하는 것이다. Remon-HC 모니터의 구조는 그림 4와 같다.

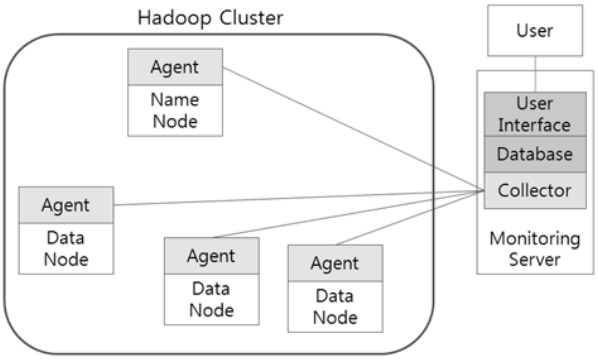


그림 4. Remon-HC 구조
Fig. 4. Remon-HC Architecture.

그림 4에서 Hadoop 클러스터를 구성하는 모든 노드에 Agent가 존재한다. Agent는 노드의 상태 정보와 Hadoop의 작업 정보를 수집하여 Collector로 전송하는 기능을 제공한다. Collector는 Agent로부터 데이터를 받은 즉시, 정보를 분석하고 데이터베이스에 저장하는 기능을 제공한다. 그리고 사용자는 사용자 인터페이스를 통해 Hadoop 클러스터의 정보를 확인할 수 있다.

Agent는 Hadoop 클러스터를 구성하는 모든 노드에서 실행되며 노드의 상태 정보, Job Tracker 로그 파일, Task Tracker 로그 파일을 수집한다. Agent 설계 시 고려할 사항은 Hadoop 작업에 지연이 발생하지 않도록 오버헤드를 최소화해야 하며, 실시간 모니터링을 위해 초 단위의 주기로 정보를 수집해야 한다.

Agent의 수행 과정 알고리즘은 그림 5와 같다.

그림 5에서 Agent는 Collector에 접속을 하고 모니터링 주기를 전송 받는다. 전송 받은 모니터링 주기마다 노드의 상태 정보부터 수집한다. 그리고 Hadoop의 로그 폴더에서 Job Tracker 로그 파일이나 Task Tracker 로그 파일이 있다면 해당 파일을 수집하고 새로운 메시지가 있다면 노드의 상태 정보와 함께 Collector로 전송

```
Information: String
Period: Integer

IF Access to Collector, and succeed
  THEN
    Period= Monitoring period getting from Collector
    FOR Each Period Runs
      Information = State of Node
      IF is Job Tracker log file
        THEN
          IF is new message
            THEN
              Information= Information+ new message
          END IF
      END IF
      IF is Task Tracker log file
        THEN
          IF is new message
            THEN
              Information= Information+ new message
          END IF
      END IF

      Send Information to Collector

    END FOR
  ELSE
    Exit Agent
END IF
```

그림 5. Agent의 알고리즘
Fig. 5. Algorithm of the Agent.

```
Split_Data: Array(node state, job info, task info, history info)
Result: String

IF Monitoring period in database
  THEN
    FOR not exit message
      IF Connection of Agent
        THEN
          Send monitoring period to Agent
      END IF
      IF Receive monitoring information to Agent
        THEN
          Split_Data = Split monitoring information
          IF has Split_Data[node state]
            THEN
              Result: Parsing Split_Data[node state]
              Insert to database for Result
          END IF

          … … (job, task and history info)

      END IF
      IF is changed monitoring period
        THEN
          Send monitoring period to all Agent
      END IF
    END FOR
  ELSE
    Exit Collector
END IF
```

그림 6. Collector의 알고리즘
Fig. 6. Algorithm of the Collector.

한다. 만약, 새로운 정보가 없다면 노드의 상태 정보만 전송한다.

Collector는 별도의 모니터링 서버에 위치하고, Hadoop 클러스터의 모든 노드에서 동작하는 Agent와 통신한다. 또한 사용자에게 실시간으로 모니터링 정보를 제공하기 위해 Agent에서 보낸 정보를 즉시 분석하여 데이터베이스에 저장한다. Collector의 수행 과정 알고리즘은 그림 6과 같다.

그림 6에서 Collector는 실행과 동시에 데이터베이스에 접속하여 모니터링 주기를 전송받는다. 그리고 Agent가 접속을 하면 모니터링 주기를 전송한다. 또한 사용자가 모니터링 주기를 변경했을 경우, 변경된 모니터링 주기를 모든 Agent에 전송한다. 그리고 Agent가 주기적으로 보내는 정보를 분석한다. 분석 단계는 노드 상태 분석, Job Tracker 로그 파일 분석, Task Tracker 로그 파일 분석, Job History 로그 파일 분석으로 나뉜다. 노드의 상태 정보에 대한 분석은 Agent 구현시, 노드 상태 수집 모듈에 명시된 형태로 분석하고, 분석이 완료된 정보는 곧바로 데이터베이스에 저장 되거나 업데이트된다.

## 2. Remon-HC 구현

본 논문에서 제안하는 Remon-HC 모니터 구현은 Agent, Collector, 데이터베이스 구축, User Interface로 구분된다. Agent와 Collector는 C언어로 구현하고, 데이터베이스는 MySql을 사용한다. Agent-Collector간의 통신을 위해 epoll 라이브러리를 사용한다. 그리고 User Interface는 운영체제에 영향을 받지 않고 모니터링이 가능하도록 Flash로 구현하여 웹 브라우저에서 확인할 수 있다. Flash는 MySql 데이터베이스에 직접 연동할 수 없기 때문에 php를 사용하여 MySql에 연동하고, 그 결과를 Flash로 전송한다. Remon-HC 모니터 구현 환경은 표 1과 같다.

표 1. Remon-HC의 구현 환경
Table 1. Implementation environment of the Remon-HC.

| 사용 툴 및 언어 | 버전 | 구동 운영체제 |
|---|---|---|
| gcc | gcc 4.5.1 | Linux Kernel 2.6.32(Ububtu) |
| POSIX epoll | - | |
| Mysql, PHP, Apache | 5.1, 5.3, 2.2 | |
| Flash | CS4 | WindowsXP |

## 3. Remon-HC 모니터 동작

Remon-HC 모니터의 동작은 User Interface의 각 항목을 통해 구체적으로 설명한다. User Interface는 Hadoop 클러스터 정보, 노드 상태 정보, Hadoop 작업 정보, 모니터 환경 설정으로 구성된다.

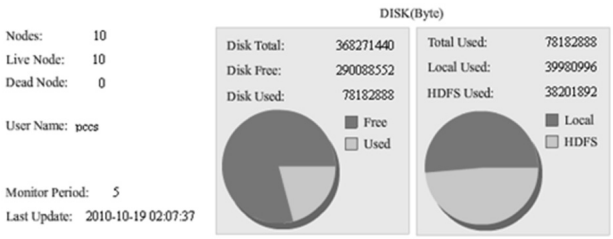


그림 7. Hadoop 클러스터 정보
Fig. 7. Information of Hadoop cluster.

### 가. Hadop 클러스터 정보

Hadoop 클러스터 정보에서는 Hadoop 클러스터에서 사용 가능한 디스크 용량과 현재 HDFS에서 사용 중인 디스크 용량을 확인할 수 있다. 또한, Hadoop 클러스터 정보에는 클러스터의 총 노드 수와 Live 노드, Dead 노드 수 등을 파악할 수 있다. Hadoop 클러스터 정보 출

력 화면은 그림 7과 같다.

### 나. 노드 상태 정보

노드 상태 정보에서는 전체 노드의 상태를 이미지로 표현하며, 특정 노드를 선택 시, CPU나 메모리 등의 정보를 시간의 흐름에 따라 그래프로 표현한다. 노드의 상태 정보는 그림 8과 같이 출력한다.

노드 상태 정보에서는 그림 8과 같이 현재 모니터링 중인 모든 노드들이 이미지로 표현되고, Host명이 출력된다. 노드의 이미지는 CPU 사용률에 따라 녹색에서 적색으로 변경되며, 특정 노드의 이미지 위에 마우스를 올리면 마지막으로 정보가 업데이트된 시간, IP, CPU와 메모리 사용률이 팝업 화면으로 나타난다.

노드 상태 정보에서 특정 노드를 선택하면 그림 9와 같이 해당 노드에 대한 세부 정보를 확인 할 수 있다.

그림 9에서 확인할 수 있는 정보로는 사용자 이름, IP, 메모리, 디스크, 현재 실행중인 프로세스 및 자바 프로세스가 있다. 또한 시간에 따른 사용률 그래프로 표현되는 정보는CPU 사용률, 메모리 사용률, 네트워크 트래픽이 있다.

### 다. Hadoop 작업 정보

Hadoop 작업 정보에서는 Hadoop MapReduce로 수행된 모든 작업의 정보를 그림 10과 같이 확인할 수 있다. 그림 10에서 작업 리스트의 특정 작업을 선택하면 작업의 시작 시간과 종료 시간, 작업의 상태, 사용한 HDFS 용량, Map Task, Reduce Task에서 사용한 입출력 크기 등을 확인할 수 있다.

그림 10에서 특정 작업을 선택한 후, View Task를 선택하면 해당 작업을 수행한 Task 들이 타임라인 형태로 그림 11과 같이 나타난다.

그림 11에서 Map Task는 청색으로 표시되고 Reduce Task는 적색, Killed Task는 회색으로 표시된다. 또한, 특정 Task 위에 마우스 커서를 올리면 해당 Task의 ID와 시작 시간, 마지막으로 업데이트된 시간, 진행률이 나타난다. 또한, 작업이 진행 중일 경우, 작업의 진행에 따라 실시간으로 타임라인이 변경되는 것을 확인할 수 있다. 그림 11에서 수행한 Task들이 노드 별로 정렬되어 나타난다. 이때, 하단의 버튼을 이용하여 CPU 사용률이나 메모리 사용률, 네트워크 사용률을 그림 12와 같이 그래프로 볼 수 있다.

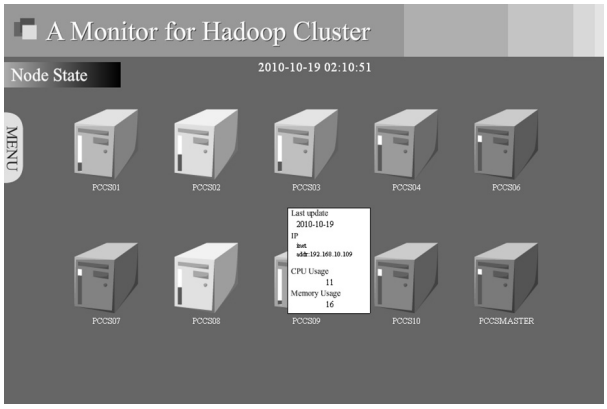


그림 8. 노드 상태 정보
Fig. 8. Information of nodes state.

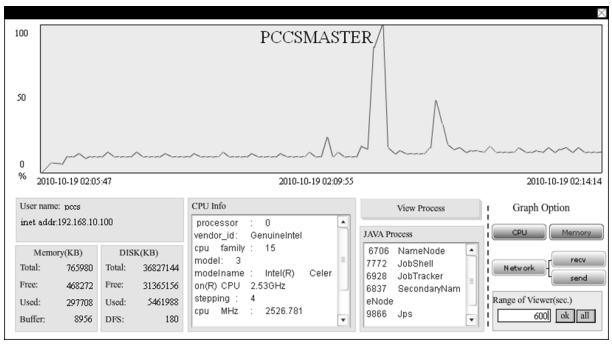


그림 9. 노드의 세부 정보
Fig. 9. Detailed information of a node.


그림 10. Hadoop 작업 정보
Fig. 10. Information of Hadoop jobs.

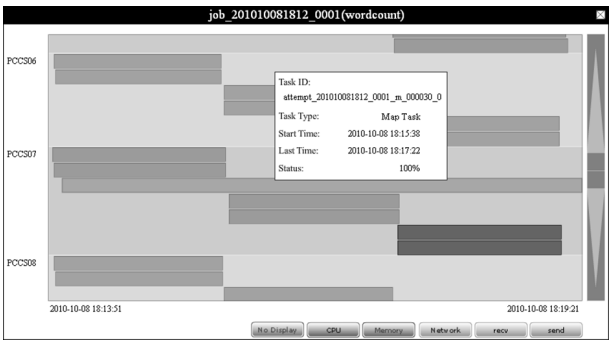


그림 11. 작업의 세부 정보
Fig. 11. Detailed information of a Hadoop job.

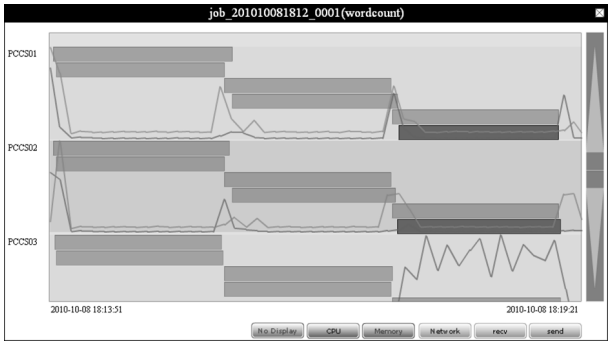


그림 12. Task와 노드 오버헤드의 일괄적인 표현
Fig. 12. Representation in a lump of Task and nodes overhead.

## Ⅳ. 성능 평가 및 결과 분석

본 장에서는 제안한 Remon-HC 모니터를 이용하여 Hadoop 클러스터에서 발생한 문제점을 발견하고, 성능 개선 효과가 있음을 검증한다. 또한 기존의 Hadoop 클러스터 모니터인 Chukwa와 성능을 비교하고 분석한다.

### 1. 실험 환경

성능평가를 위해 Hadoop 클러스터와 모니터링 서버를 구축한다. Hadoop 클러스터의 환경은 표 2와 같고, 모니터링 서버의 환경은 표 3과 같다.

성능평가를 위해 모니터를 수행하면서 Hadoop 애플리케이션의 작업 수행시간을 측정하고 그 결과를 비교한다. 수행할 애플리케이션은 Wordcount[8]로써, HDFS에 저장된 파일을 읽어서 블록 단위로 나누고, 각 블록에 대해서 단어 빈도수를 계산한다. Wordcount의 입력 파일은 Randomtextwriter로 생성된 3GB의 텍스트 파일이며, 모든 실험에서 동일한 파일을 사용한다.

표 2. Hadoop 클러스터의 환경
Table 2. Environment of Hadoop cluster.

| | | |
|---|---|---|
| 노드 수 | 네임 노드 | 1 개 |
| | 데이터 노드 | 9 개 |
| 노드 성능 | CPU 성능 | Intel 2.53 Ghz |
| | Memory 용량 | 768 MB |
| 운영체제 | Linux Kernel 2.6.32(Ubuntu server) | |
| Hadoop | Hadoop-0.18.3 | |
| 네트워크 대역폭 | 100 Mbps | |

표 3. 모니터링 서버의 환경
Table 3. Environment of a monitoring server.

| | |
|---|---|
| CPU 성능 | AMD 2.92 Ghz(Quad core) |
| Memory 용량 | 4096 MB |
| 운영체제 | Linux Kernel 2.6.32(Ubuntu server) |
| MySql, PHP, Apache web server | MySql-5.1, PHP-5.3, Apache-2.2 |

### 2. Remon-HC 모니터의 활용

Remon-HC 모니터를 이용하여 Hadoop의 MapReduce 작업을 수행하면서 발견한 비효율적인 부분을 Hadoop 설정을 수정한다. 그리고 Hadoop 클러스터의 성능을 개선한 사례를 보인다.

Wordcount를 수행한 결과는 그림 13과 같다.

그림 13에서 PCCS02 노드와 PCCS03 노드의 마지막 Task가 Kill 된 것을 볼 수 있다. 이 현상의 원인은 다른 노드에서 실행중인 Task를 유휴 노드에서 동일한 Task를 실행한 것이다. 즉, 동일한 Task 수행 중, 먼저 완료된 Task가 있다면 다른 Task는 Kill 되는 것이다. 이는 Hadoop의 기본 설정이지만, 사용자가 클러스터의 환경에 맞게 변경할 수 있다. 또한, 하나의 노드에서 2

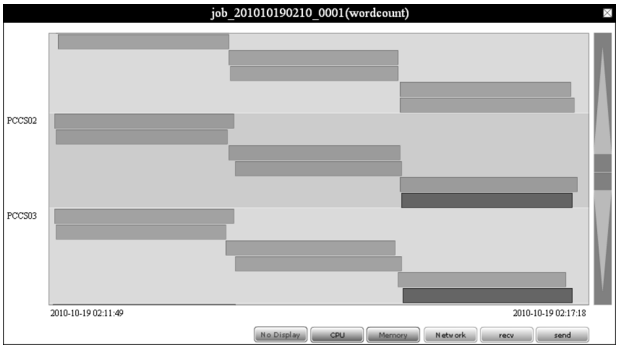


그림 13. Wordcount 수행 결과
Fig. 13. Result of the Wordcount execution.

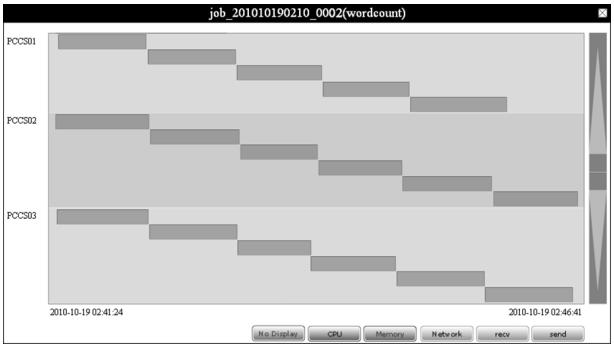


그림 14. 개선된 클러스터 환경에서 Wordcount 수행 결과
Fig. 14. Result of the Wordcount execution in improved cluster environment.

개의 Task가 동시에 실행되는 것을 볼 수 있는데, 이러한 사항도 Hadoop의 기본 설정이다. 본 실험에서 사용된 모든 노드의 CPU는 싱글 코어이기 때문에 멀티스레딩에 불리하다. 따라서 이 두 가지의 설정을 변경함으로써 성능 개선을 도모할 수 있으며, 유휴 자원을 확보할 수 있다.

Hadoop의 환경설정을 변경하고, Wordcount를 수행한 결과는 그림 14와 같다.

그림 14에서 기존의 클러스터보다 12초 빠른 수행시간을 보이고 있다. 또한 작업이 먼저 완료된 PCCS01 노드는 유휴자원이 된다.

### 3. 모니터 오버헤드

Hadoop의 MapReduce 작업 수행시간을 측정하여 비교함으로써 제안한 Remon-HC 모니터의 성능을 평가한다. 즉, 모니터링하지 않을 때의 Wordcount 수행시간과 Chukwa로 모니터링 할 때의 Wordcount 수행시간, 그리고 제안한 Remon-HC로 모니터링 할 때의 Wordcount 수행시간을 측정하여 비교한다.

실험 횟수는 5회이며, 그 결과는 그림 15와 같다.

그림 15를 살펴보면 모니터링하지 않을 때인 Non-모니터의 Wordcount 수행시간이 가장 우수하다. 또한, Chukwa와 Remon-HC로 모니터링 할 때의 Wordcount 수행시간이 Non-모니터보다 높음을 볼 수 있다. 따라서 Wordcount 수행시 모니터링을 위해서는 오버헤드가 발생함을 알 수 있다. 한다. 그리고 Remon-HC로 모니터링 할 때의 Wordcount 수행시간이 Chukwa로 모니터링 할 때 보다 성능이 우수함을 알 수 있다.

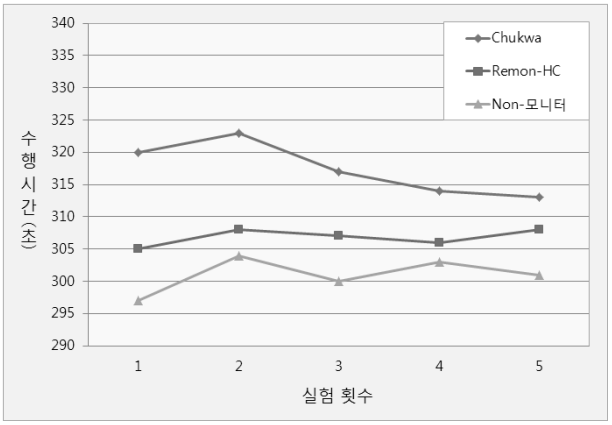


그림 15. Wordcount 수행시간 비교
Fig. 15. Execution time comparison of Wordcount.

## V. 결 론

클라우드 컴퓨팅을 위한 Hadoop은 플랫폼은 HDFS와 MapReduce 프로그래밍 모델을 제공한다. Hadoop을 이용하여 클러스터를 구축하고 운영할 때, Hadoop클러스터를 보다 쉽게 관리하기 위해 모니터를 사용한다.

기존의 모니터는 클러스터를 구성하고 있는 노드들의 상태 정보만을 수집하기 때문에 Hadoop의 작업 정보를 모니터링 할 수 없는 문제점이 있다. 이 문제점 해결하기 위해 Hadoop의 작업정보를 수집할 수 있는 모니터가 제안되었지만, 수집한 데이터를 분석하기 위한 지연시간이 발생하여 실시간으로 모니터링을 할 수 없는 또 다른 문제점이 발생하였다. 따라서 노드의 이상이나 작업 실패에 대한 빠른 조치를 위해 실시간으로 노드 정보와 작업 정보를 수집할 수 있는 모니터가 필요하다.

따라서 본 논문에서는 Hadoop 클러스터의 노드 정보와 작업 정보를 실시간으로 수집할 수 있는 Remon-HC 모니터를 제안하였다. 이 모니터는 Hadoop클러스터의 노드 정보와 작업정보를 수집을 하는 Agent와 수집된 정보를 분석하고 데이터베이스에 저장하는 Collector로 구성된다. 또한 Hadoop 클러스터에 포함되지 않는 새로운 노드에 Collector를 위치시킴으로써 분석과정에서 발생하는 오버헤드가 Hadoop의 작업에 영향을 미치지 않도록 하였다. 그리고 Remon-HC 모니터를 구현하여 실험적 클러스터에 적용함으로써, Dead노드 발생을 실시간으로 파악하고, Hadoop 작업수행 시 비효율적인 작업 과정을 발견하여 수정함으로써 작업수행시간을 단축시킬 수 있었다.

## 참 고 문 헌

[1] A. Kimball, S. Michels-Slettvet and C.Biscigilia, “Cluster Computing for Web-Scale Data Processing,” *Proceeding of the 39th SIGCSE technical symposium on Computer science education*, Portland, Oregon, pp. 116-120, March 2008.

[2] Hadoop, *http://hadoop.apache.org*

[3] D. Nurmi, R. Wolski, C. Grzegorczyk, G. Obertelli, S. Soman, L. Youseff, and D. Zagorodnov, “The Eucalyptus open-source cloud-computing system,” *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing*

*and the Grid table of contents,* pp. 124-131, 2009.
[4] Amazon Elastic Compute Cloud, *http://aws.amazon.com/ec2*
[5] J. Boulon, A. Konwinski, R. Qi, A. Rabkin, E. Yang and M. Yang, "Chukwa: A large-scale monitoring system," *Proceeding of international conference on Cloud Computing and Its Applications*, pp. 1-5, Oct. 2008.
[6] J. Tan, X. Pan, S. Kavulya, R. Gandhi and P. Narasimhan, "Mochi: Visualizing Log-Anlaysis Based Tools for Debugging Hadoop," *In USENIX Workshop on Hot Topics in Cloud Computing(HotCloud)*, SanDiego, CA, Jun. 2009.
[7] S. Ghemawat, H. Gobioff, S.T. Leung, "The Google file system," *ACM SIGOPS Operating Systems Review*, Vol. 37, No. 5, pp. 29-43, Dec. 2003.
[8] Tae Hoon Keum, Won Joo Lee, Chang Ho Jeon, "A Performance Analysis Based on Hadoop Application's Characteristics in Cloud Computing," Journal of The Korea Society of Computer and Information, Vol. 15, No. 5, pp. 49-56, May 2010.

## 저 자 소 개

**금 태 훈**(정회원)
2009년 경일대학교 컴퓨터공학과 학사.
2011년 한양대학교 컴퓨터공학과 석사.
2012년 현재 LG전자 MC 사업본부 연구원
<주관심분야 : 클라우드 컴퓨팅, Grid 컴퓨팅>

**이 원 주**(정회원)-교신저자
1989년 한양대학교 전자계산학과 학사.
1991년 한양대학교 컴퓨터공학과 석사.
2004년 한양대학교 컴퓨터공학과 박사.
2012년 현재 인하공업전문대학 컴퓨터정보공학부 컴퓨터정보과 부교수.
<주관심분야 : 병렬처리시스템, 모바일컴퓨팅, Grid 컴퓨팅, 클라우드 컴퓨팅>

**전 창 호**(평생회원)
1977년 한양대학교 전자공학과 학사.
1982년 Cornell University, 컴퓨터공학과 석사.
1986년 Cornell University, 컴퓨터공학과 박사.
1977년~1979년 전자통신연구소 연구원.
현재 한양대학교 ERICA 캠퍼스 컴퓨터공학과 교수.
<주관심분야 : 병렬처리시스템, 모바일컴퓨팅, Grid 컴퓨팅, 클라우드 컴퓨팅>