# 동적인 하둡 응용 모니터링 서비스를 위한 시계열 데이터 관리 방안에 관한 연구

곽재혁*, 최지은*, 김상완*, 변은규*
*한국과학기술정보연구원
e-mail:jhkwak@kisti.re.kr

# A Study on Time-series Data Management Scheme for Dynamic Hadoop Application Monitoring Service

Jae-Hyuck Kwak*, Jieun Choi*, Sangwan Kim*, Eun-kyu Byun*
*Korea Institute of Science and Technology Information

요 약

본 논문에서는 리눅스에서 제공하는 성능 분석 도구들을 활용하여 사용자가 원하는 모니터링 매트릭을 동적으로 등록하고 모니터링할 수 있는 확장 가능한 하둡 응용 모니터링 서비스의 시계열 데이터 관리 방안을 다룬다. 본 논문에서는 이를 위해서 시계열 데이터를 위한 관계형 데이터베이스인 TimeScaleDB를 사용하였으며 동적으로 변경가능한 모니터링 매트릭 데이터가 하이퍼테이블의 관리를 통해서 구조화된 밀집 데이터 형태로 효율적으로 관리될 수 있음을 제시하였다.

## 1. 서론

하둡은 하둡 2.0부터 얀을 통해서 자원 관리를 수행한다. 하둡을 통해 응용을 실행하면 얀의 리소스매니저를 통해서 자원을 할당받고 응용을 수행하며 리소스매니저가 제공하는 REST API를 통해서 하둡 응용이 사용하는 자원의 상태를 모니터링하게 된다. 그러나 REST API가 제공하지 않는 정보는 제공받을 수 없는 한계를 가지고 있다. 이를 극복하기 위해서 리눅스에서 제공하는 성능 분석 도구들을 활용하여 사용자가 원하는 모니터링 매트릭을 동적으로 등록하고 모니터링할 수 있는 하둡 응용 모니터링 서비스가 현재 개발 중에 있다.[1]

하둡 응용 모니터링 데이터는 기본적으로 시계열 데이터의 특성을 가지므로 시계열 데이터를 적절하게 관리할 수 있는 방안이 필요하다. 본 논문에서는 이를 위해서 시계열 데이터를 위한 관계형 데이터베이스인 TimeScaleDB를 사용하였으며 동적인 모니터링 매트릭 데이터가 TimeScaleDB를 기반으로 어떤 방식으로 관리될 수 있는지를 제시하였다.

## 2. 동적인 하둡 응용 모니터링 서비스 구조

본 논문과 관련하여 개발되고 있는 동적인 하둡 응용 모니터링 서비스는 리눅스에서 제공하는 pidstat, perf등의 성능분석도구를 사용하여 모니터링 메트릭을 추출하며 이것을 Ambari[2]라고 하는 빅데이터 운영관리시스템을 기반으로 확장하고 있다. 리눅스 성능 분석 도구들은 기본적으로 프로세스 아이디(pid)를 기반으로 하고 있다. 하둡 응용이 얀에서 실행되면 어플리케이션 아이디(appid)와 컨테이너 아이디(containerid)가 생성되는데 컨테이너는 하둡 응용이 실행되는 최소 단위이며 프로세스 아이디와 직접적으로 매칭된다. 따라서 어플리케이션 아이디, 컨테이너 아이디, 프로세스 아이디는 그림 1와 같이 매칭될 수 있으며 매칭 정보는 jps나 ps등을 통해서 추출할 수 있다.

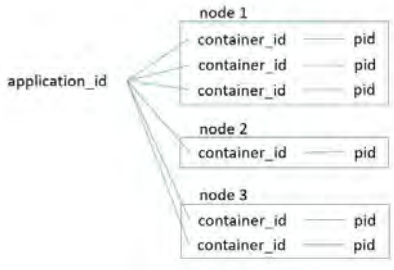


그림 1 appid, containerid, pid 관계

동적인 하둡 응용 모니터링 서비스의 전체적인 구조는 그림 2과 같다. 사용자는 Ambari Web을 통하여 사용자가 원하는 모니터링 매트릭과 파서를 플러그인 레포지토리에 등록할 수 있다. 이후에 YARNJobMonitorMaster는 타임스탬프(Tn)를 생성하고 REST API[3]를 통해서 얀의 리소스매니저로부터 appid와 리소스매니저가 제공하는 정보를 가져와서 [Tn, appid, rm_data]를 데이터베이스에 저장한다. 또한, YARNJobMonitorWorker에게 [Tn, appid]를 전달한다. YARNJobMonitorWorker는 appid를 이용하여

프로세스 아이디, 노드, 컨테이너 아이디를 추출하고 플러그인 레포지토리에 등록된 파서를 이용하여 모니터링 매트릭 데이터를 구한뒤에 [Tn, hostname, appid, containerid, pid, monitor_data]를 데이터베이스에 저장한다. Ambari Web은 데이터베이스로부터 타임스탬프 Tn에 기반하여 모니터링 데이터를 표출하게 된다.

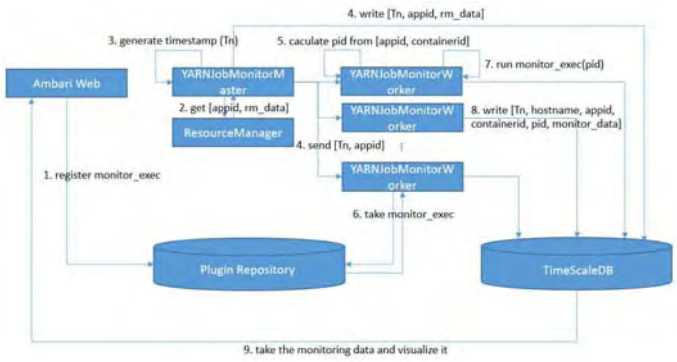


그림 2 동적인 하둡응용 모니터링 서비스 구조

### 3. TimeScaleDB를 사용한 모니터링 데이터 저장 방안

TimeScaleDB[4][5]는 PostgreSQL 데이터베이스를 확장하여 시계열 데이터에 특화된 데이터베이스로서 개발되었으며 하이퍼테이블 개념을 사용하고 있다. 사용자는 하이퍼테이블 수준에서 SQL 연산을 실행하며 하이퍼테이블은 내부적으로 시간 간격과 주요 키의 2차원으로 전체 데이터가 나누어진 청크 단위로 내부 테이블을 관리하며 삽입 연산 위주의 시계열 데이터에 적합한 관리 구조를 가진다.

본 논문에서 언급한 하둡 응용 모니터링 서비스는 동적으로 모니터링 매트릭을 등록하거나 삭제할 수 있기 때문에 이를 관계형 데이터베이스 테이블로 구현하느냐가 중요하다. ALTER TABLE문을 통해서 컬럼을 동적으로 변경하는 방식을 생각해볼 수 있는데 ADD COLUMN문을 통해서 컬럼을 추가하는 경우 NULL값으로 채워지는 컬럼들이 불필요하게 발생할 수 있으며 DROP COLUMN 문을 통해서 컬럼을 삭제하는 경우 컬럼 데이터가 불필요하게 삭제되는 문제가 발생할 수 있다.

동적인 하둡 모니터링 서비스에서 관리자가 모니터링 매트릭을 동적으로 추가하거나 삭제하는 일은 빈번하게 발생하지 않는다는 점을 가정하여 본 논문에서는 하둡 응용 모니터링 서비스에 새로운 모니터링 매트릭이 등록될 때마다 하이퍼테이블을 새로이 생성하는 방법을 제안한다. 하이퍼테이블명은 선후 관계를 파악하기 위한 목적으로 [seqeuence number].apptable.[starting timestamp]로 하였다. 모니터링 매트릭이 변경되는 시점에서 새로이 하이퍼테이블을 생성하므로 각각의 하이퍼테이블은 구조화된 밀집 데이터를 가지므로 관계형 데이터베이스에 적합한 형태로 볼 수 있다.

만약, 2018년 10월 1일 서비스 이후에 최초로 등록된 모니터링 매트릭이 CPU, MEMORY라고 가정하면 하이퍼테이블은 다음과 같이 생성된다.

```
CREATE TABLE "0.apptable.20181001" (
 time TIMESTAMP WITHOUT TIME ZONE NOT NULL,
 hostname TEXT NOT NULL,
 appid TEXT NOT NULL,
 containerid TEXT NOT NULL,
 pid TEXT NOT NULL,
 cpu NUMERIC NOT NULL,
 memory NUMERIC NOT NULL,
);
SELECT create_hypertable('0.apptable.20181001', 'time');
CREATE INDEX ON "0.apptable.20181001" (appid, time desc);
CREATE INDEX ON "0.apptable.20181001" (hostname, time desc);
```

만약, 2018년 10월 31일에 모니터링 매트릭으로 IO를 추가했다고 가정하면 다음과 같이 새로운 하이퍼테이블이 생성되어 사용된다.

```
CREATE TABLE "1.apptable.20181031" (
 time TIMESTAMP WITHOUT TIME ZONE NOT NULL,
 hostname TEXT NOT NULL,
 appid TEXT NOT NULL,
 containerid TEXT NOT NULL,
 pid TEXT NOT NULL,
 cpu NUMERIC NOT NULL,
 memory NUMERIC NOT NULL,
 io NUMERIC NOT NULL,
);
SELECT create_hypertable('1.apptable.20181031','time');
CREATE INDEX ON "1.apptable.20181031" (appid, time desc);
CREATE INDEX ON "1.apptable.20181031" (hostname, time desc);
```

모니터링 데이터가 데이터베이스에 저장되어 있다면 Ambari Web은 데이터베이스에 저장된 데이터를 가져와서 표출하면 된다. appid에 기반하여 응용별로 그래프를 표출할 수도 있고 hostname에 기반하여 노드별로 그래프를 표출할 수도 있다. appid에 기반하여 응용별로 그래프를 표출하는 경우를 보면, 먼저 동일한 appid에 해당하는 모니터링 데이터를 저장하고 있는 하이퍼테이블을 찾는다. 모니터링 매트릭은 하둡 응용이 실행되지 않는 상황에서만 등록이 가능하기 때문에 appid에 해당하는 모니터링 데이터는 하나의 하이퍼테이블만 검색하면 된다. appid의 시작시간 및 종료시간과 하이퍼테이블명에 포함된 starting timestamp로부터 appid의 모니터링 데이터가 존재하는 하이퍼테이블을 찾을 수 있으며 하이퍼테이블을 찾았으면 하이퍼테이블로부터 appid와 관련된 모니터링 데이터를 쿼리하기만 하면 된다.

hostname에 기반하여 노드별로 그래프를 표출하는 경우를 보면, 모니터링 매트릭이 등록되고 하이퍼테이블이 추가된 이후에 hostname에 해당하는 모니터링 데이터는 복수개의 하이퍼테이블에 존재할 수 있다. 따라서, 표출하기를 원하는 전체 시간 구간에 대해서 먼저 입력받아야 한다. 이 값을 하이퍼테이블명에 포함된 starting timestamp와 비교하면 동일한 hostname을 포함하고 있는 하이퍼테이블을 찾을 수 있으며 복수 개의 하이퍼테이블이 검색될 수 있다. 이후에는 각각의 하이퍼테이블로부터 hostname에 기반한 모니터링 데이터를 가져오면 된다. 각각의 하이퍼테이블은 timestamp가 배타적으로 존재하고 각각의 하이퍼테이블명은 sequece number를 포함하고 있으므로 순서에 맞게 각각의 하이퍼테이블에서 쿼리한 결과를 단순히 시간 순으로 나열하여 그래프로 표출하기만

하면 되며 별도의 조인 연산을 필요하지 않을 것이다.

## 3. 결론 및 향후 계획

본 논문에서는 동적인 하둡 응용 모니터링 서비스를 위해 요구되는 시계열 데이터의 관리 방안에 대해서 살펴보았다. 본 논문에서는 TimeScaleDB를 시계열 데이터를 위한 관계형 데이터베이스를 사용하였으며 모니터링 매트릭의 변경 시점에서 하이퍼테이블의 동적인 관리를 통해 시계열 모니터링 데이터를 효율적으로 관리할 수 있는 방안을 제시하였으며 모니터링 데이터 표출을 위해서 응용별, 노드별로 시계열 모니터링 데이터가 어떻게 쿼리될 수 있는지를 제시하였다.

※ 이 논문은 2018년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원(No. R0190-18-2012, 빅데이터 처리 고도화 핵심기술개발 사업 총괄 및 고성능 컴퓨팅 기술을 활용한 성능 가속화 기술 개발)과 한국과학기술정보연구원 주요사업의 지원을 받아 수행된 연구임

## 참고문헌

[1] 곽재혁, 김상완, 변은규, “리눅스 성능 분석 도구를 활용한 암바리 기반 하둡 응용 모니터링 서비스 설계에 관한 연구”, 2018 한국컴퓨터종합학술대회(KCC2018), 2018.

[2] Apache Ambari, http://ambari.apache.org

[3] ResourceManager REST API's, https://hadoop.apache.org/docs/r2.7.3/hadoop-yarn/hadoop-yarn-site/ResourceManagerRest.html

[4] TimeScale, http://www.timescale.com

[5] Time-series Data: Why (and how) to use a relational database instead of NoSQL, https://blog.timescale.com/time-series-data-why-and-how-to-use-a-relational-database-instead-of-nosql-d0cd6975e87c

[6] Apache Hadoop, http://hadoop.apache.org

[7] T.White, Hadoop: The Definitive Guide, OREILLY, 2010.