Apache Kudu와 Impala를 활용한 Lambda Architecture 설계

황윤영*, 이필원*, 신용태**
*숭실대학교 컴퓨터학과
**숭실대학교 컴퓨터학부
doublewhy@soongsil.ac.kr, pwlee@soongsil.ac.kr, shin@ssu.ac.kr

Lambda Architecture Design using Apache Kudu and Impala

Yun-Young Hwang, Pil-Won Lee, Yong-Tae Shin
Soongsil University, Science of Computer

요 약

데이터의 양은 기술의 발전으로 발생하는 크게 증가하였고 다양한 빅데이터 처리 플랫폼이 등장하고 있다. 이 중 가장 널리 사용되고 있는 플랫폼이 Apache 소프트웨어 재단에서 개발한 Hadoop이며, Hadoop은 IoT 분야에도 사용된다. 그러나 기존에 Hadoop 기반 IoT 센서 데이터 수집 분석 환경은 Hadoop의 코어 프로젝트인 HDFS의 Small File로 인한 네임노드의 과부하 문제와 Import된 데이터의 Update나 Delete가 불가능하다는 문제가 있다. 본 논문에서는 Apache Kudu와 Impala를 활용해 Lambda Architecture를 설계한다. 제안하는 Architecture는 IoT 센서 데이터를 Cold-Data와 Hot-Data로 분류해 각 성격에 맞는 스토리지에 저장하고 Batch를 통해 생성된 Batch-View와 Apache Kudu와 Impala를 통해 생성된 Real-time View를 활용해 기존 Hadoop 기반 IoT 센서 데이터 수집 분석 환경의 문제를 해결하고 사용자가 분석된 데이터에 접근하는 시간을 단축한다.

1. 서론

데이터의 발생량은 5G의 등장으로 초고속, 초저지연을 이용한 새로운 IoT 기술이 등장하고 발전하면서 폭발적으로 증가하고 있다. 다양한 빅데이터 처리 플랫폼이 이를 처리하기 위해 등장하고 있다. Hadoop은 이 중 가장 널리 사용되고 있는 플랫폼 중 하나로 Apache 소프트웨어 재단에서 개발했다. Hadoop은 빅데이터를 수집, 저장, 처리, 분석, 시각화하는 다양한 서브 프로젝트를 프레임워크로 제공한다. Hadoop의 코어 프로젝트인 HDFS(Hadoop Distributed File System)는 블록 기반의 대용량 데이터 저장소로 최소 64MB에서 256MB 크기의 블록 단위에 데이터를 저장하기 때문에 설정된 블록의 사이즈를 최대한 활용해야 효율성이 좋아진다.

그러나 작은 단위의 데이터를 지속적으로 생성하는 IoT 센서 데이터 수집 분석 환경의 경우 HDFS에 구성된 최소 크기의 블록만큼 데이터가 생성되기 전에 저장되는 Small File 문제로 인해 네임노드에 과부하가 발생해 전체적인 시스템의 성능을 저하시키는 문제가 있다.[1] HDFS는 블록에 파일 형식으로 데이터를 저장하기 때문에 Import된 데이터의 Update나 Delete가 불가능하다. Apache Kudu는 이와 같은 문제를 해결하기 위해 개발되었다.

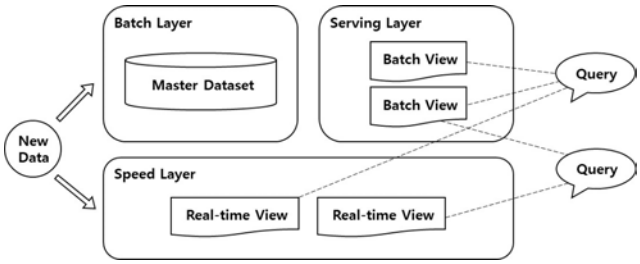
제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 Hadoop 기반의 IoT 센서 데이터 수집 분석 환경에서 발생하는 Small File 문제를 해결한다. 제안하는 Architecture는 Hadoop에서 불가능한 Import된 데이터의 수정 문제를 Apache Kudu와 Impala를 활용하여 해결한다. 갱신주기가 짧으며 크기가 작은 실시간 데이터와 갱신주기가 긴 저장되어 있는 대용량 데이터를 빠르게 분석하는 환경을 제공한다.

본 논문의 구성은 다음과 같다. 2장에서는 Lambda Architecture의 정의와 특징, Apache Kudu와 Apache Impala에 대해 살펴본다. 3장에서는 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture를 설계한다. 4장에서는 결론 및 향후 연구 방향을 제시한다.

2. 관련 연구

2-1. Lambda Architecture

Lambda Architecture는 오래된 데이터를 보관하는 배치(Batch) 테이블과 실시간 데이터를 가진 실시간 테이블을 JOIN하여 결과값을 얻을 수 있도록 구성한 Architecture이다.[2] [그림 1]은 Lambda Architecture의 구조를 나타낸다.

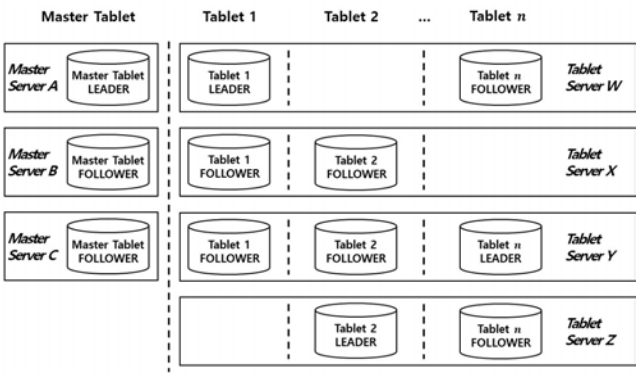


[그림 1] Lambda Architecture의 구조

Lambda Architecture는 Batch Layer, Speed Layer, Serving Layer로 구성되어 있다. Batch Layer에서는 Batch를 이용해 데이터를 미리 계산하여 저장소에 raw 데이터를 보관한다. Batch View의 데이터가 부정확할 때 저장소의 raw 데이터를 통해 복구가 가능하다. 기존의 raw 데이터는 새로운 View를 제공하고자 할 때 통계 분석이 가능하다. Speed Layer는 데이터를 실시간으로 집계해 별도의 테이블에 저장하여 Batch가 실행되는 동안 발생하는 조회에 대한 공백 문제를 해결한다. Serving Layer는 Batch Layer 및 Speed Layer의 출력을 저장한다. 클라이언트는 Serving Layer에 저장된 데이터를 조회하기 때문에 빠른 응답이 가능하다.

2-2. Apache Kudu

Apache Kudu는 Apache 소프트웨어 재단에서 개발한 컬럼 지향 데이터 스토리지이다. Apache Kudu는 Hadoop 기반의 프레임워크 대부분과 호환되며 Hadoop의 범용 하드웨어 사용성, 확장성, 데이터 가용성 보증의 특성을 지원한다. Apache Kudu는 블록 기반의 스토리지가 아닌 NoSQL OLAP 데이터베이스로 HDFS에서 불가능한 Update와 Delete 명령문을 지원한다. Apache Kudu는 데이터를 칼럼 기반으로 저장하여 특정 칼럼만 읽을 때는 디스크에서 읽는 데이터의 양을 줄여 성능을 높인다. Apache Kudu는 일반 DBMS처럼 Primary Key를 제공하며, Primary Key는 내부적으로 B+트리로 저장되어 대규모 데이터에서 빠르게 원하는 데이터에 접근한다. Apache Kudu는 데이터 저장소 역할만 하는 플랫폼으로 이를 사용하기 위해 서버가 필요하다.[3] [그림 2]는 Apache Kudu의 서버 구성을 나타낸다.

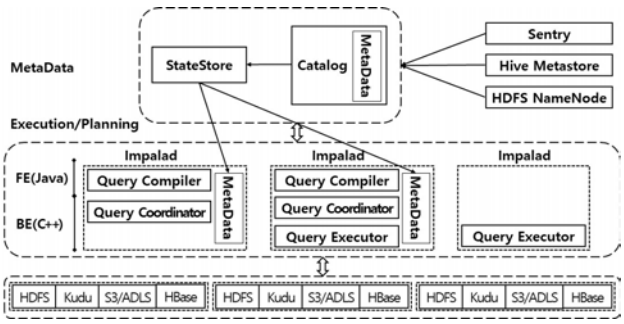


[그림 2] Apache Kudu의 서버 구성

Apache Kudu 내 데이터는 구조화되어 테이블에 저장되며, 테이블은 Tablet이라는 단위로 세분화되어 Tablet 서버에 저장된다. 스토리지 시스템은 메타데이터를 관리하는 마스터 노드와 사용자 데이터인 Tablet을 저장하는 Tablet 서버로 구성된다. Apache Kudu는 하나 이상의 마스터 노드와 Tablet 서버로 구성된다. Apache Kudu는 단순한 CRUD만 제공하기 때문에 복잡한 질의를 실행하기 위한 별도의 질의 처리기가 필요하다.

2-3. Apache Impala

Apache Impala는 HDFS을 위해 Apache 소프트웨어 재단에서 개발한 분산 병렬 질의 처리 엔진이다. Apache Impala는 스토리지에 저장되어 있는 데이터를 SQL을 통해 실시간으로 분석하는 시스템으로 스토리지 엔진이 제공하지 않는 연산을 실행한다.[4] Apache Impala는 MapReduce를 이용하지 않는 분산 질의 엔진을 통해 SQL을 실행하여 낮은 지연시간으로 결과를 제공한다. [그림 3]은 Apache



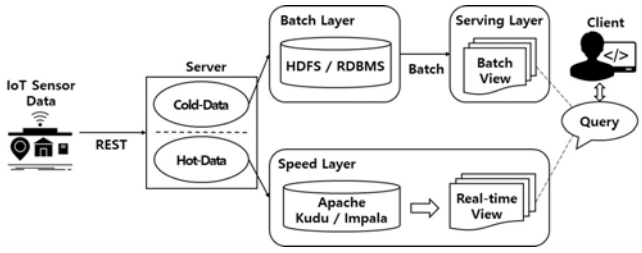
[그림 3] Apache Impala의 구성

Impala의 구조를 나타낸다.

Apache Impala는 Daemon, Catalog Service, Statestore로 구성된다. Daemon은 데이터 노드에서 실행되는 프로세스로 사용자의 요청을 수용하고, Coordinator와 Executor 역할을 한다. Catalog Service는 메타데이터의 동기화를 위한 프록시 역할을 하며 Daemon에서 직접 메타데이터를 변경하면 자동으로 동기화된다. Statestore는 Daemon의 상태를 확인하고 메타데이터를 동기화한다.

3. 제안하는 Lambda Architecture 설계

제안하는 Architecture는 기존의 HDFS 기반 IoT 센서 데이터 수집 분석 환경을 Apache Kudu와 Impala를 활용해 Lambda Architecture로 구성한다. 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 발생되는 데이터를 자주 사용되지 않고 갱신이 적은 대용량의 Cold-Data와 자주 사용되고 갱신주기가 짧고 크기가 작은 실시간 Hot-Data로 분류한다. 데이터가 크기에 따라 분류되어 각 성격에 맞는 스토리지에 저장됨으로써 HDFS의 Small File 문제를 해결할 수 있다. Speed Layer에서는 Real-time View를 끊임없이 생성해 Batch가 실행될 때 발생하는 공백 문제를 해결할 수 있다. [그림 4]는 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture의 구성을 나타낸다.



[그림 4] 제안하는 Lambda Architecture의 구성

제안하는 Architecture는 IoT 센서 데이터를 Rest 통신으로 서버에 전송하며, 서버는 이를 Cold-Data와 Hot-Data로 분류한다. Cold-Data는 HDFS에 저장되고, HDFS는 Batch를 통해 주기적으로 Batch View를 생성한다. Hot-Data는 Apache Kudu에 저장되는 동시에 누락된 데이터를 삭제하거나 갱신하여 데이터의 무결성을 보장하며 Impala를 통해 Real-time View를 생성한다. 클라이언트는 SQL 질의를 통해 Batch View와 Real-time View의 JOIN 결과를 제공받는다.

4. 결론

본 논문에서 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 Bactch View와 Apache Kudu와 Impala로 생성된 Real-time View를 통해 클라이언트가 결과까지 접근하는 시간을 단축하며, Hadoop 기반 데이터 수집 분석 환경에서 발생하는 Small File로 인한 네임노드의 과부하 문제를 해결할 수 있다. 향후 본 논문에서 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture의 구축이 필요하며, Cold-Data와 Hot-Data를 운영환경에 맞춰 자동으로 분류하는 알고리즘의 연구가 필요하다.

Acknowledgement

이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No.IITP-2019-0-00135, ICT 기반 환경 모니터링 센서 신뢰성 검증 및 평가 플랫폼)

참고문헌

[1] Bende, Sachin, and Rajashree Shedge. "Dealing with small files problem in hadoop distributed file system." Procedia Computer Science 79 pp. 1001-1012, 2016.

[2] KIRAN, Mariam, et al. Lambda architecture for cost-effective batch and speed big data processing. In: 2015 IEEE International Conference on Big Data (Big Data). IEEE, p. 2785-2792, 2015.

[3] Lipcon, Todd, et al. "Kudu: Storage for fast analytics on fast data." Cloudera, inc 28, 2015.

[4] KORNACKER, Marcel; ERICKSON, Justin. Cloudera impala: Real time queries in apache hadoop, for real. Ht Tpblog Cloudera Comblog201210cloudera-Impala-Real-Time-Queries--Apache-Hadoop--Real, 2012.