KIPS Trans. Comp. and Comm. Sys.
Vol.9, No.9 pp.207~212
ISSN: 2287-5891 (Print), ISSN 2734-049X (Online)

https://doi.org/10.3745/KTCCS.2020.9.9.207

Lambda Architecture Used Apache Kudu and Impala

Yun-Young Hwang† · Pil-Won Lee† · Yong-Tae Shin††

ABSTRACT

The amount of data has increased significantly due to advances in technology, and various big data processing platforms are emerging, to handle it. Among them, the most widely used platform is Hadoop developed by the Apache Software Foundation, and Hadoop is also used in the IoT field. However, the existing Hadoop-based IoT sensor data collection and analysis environment has a problem of overloading the name node due to HDFS' Small File, which is Hadoop's core project, and it is impossible to update or delete the imported data. This paper uses Apache Kudu and Impala to design Lambda Architecture. The proposed Architecture classifies IoT sensor data into Cold-Data and Hot-Data, stores it in storage according to each personality, and uses Batch-View created through Batch and Real-time View generated through Apache Kudu and Impala to solve problems in the existing Hadoop-based IoT sensor data collection analysis environment and shorten the time users access to the analyzed data.

Keywords : Apache Hadoop, HDFS, Apahce Kudu, Apache Impala, Lambda Architecture, IoT

Apache Kudu와 Impala를 활용한 Lambda Architecture 설계

황 윤 영† · 이 필 원† · 신 용 태††

요 약

데이터의 양은 기술의 발전으로 크게 증가하였고 이를 처리하기 위해 다양한 빅데이터 처리 플랫폼이 등장하고 있다. 이 중 가장 널리 사용되고 있는 플랫폼이 Apache 소프트웨어 재단에서 개발한 하둡이며, 하둡은 IoT 분야에도 사용된다. 그러나 기존에 하둡 기반 IoT 센서 데이터 수집 분석 환경은 하둡의 코어 프로젝트인 HDFS의 Small File로 인한 네임노드의 과부하 문제와 임포트된 데이터의 업데이트나 삭제가 불가능하다는 문제가 있다. 본 논문에서는 Apache Kudu와 Impala를 활용해 Lambda Architecture를 설계한다. 제안하는 구조는 IoT 센서 데이터를 Cold-Data와 Hot-Data로 분류해 각 성격에 맞는 스토리지에 저장하고 배치를 통해 생성된 배치뷰와 Apache Kudu와 Impala를 통해 생성된 실시간뷰를 활용해 기존 하둡 기반 IoT 센서 데이터 수집 분석 환경의 문제를 해결하고 사용자가 분석된 데이터에 접근하는 시간을 단축한다.

키워드 : Apache Hadoop, HDFS, Apache Kudu, Apache Impala, Lambda Architecture, IoT

1. 서 론

데이터의 발생량은 5G의 등장으로 초고속, 초저지연을 이용한 새로운 IoT(Internet of Things) 기술이 등장하고 발전하면서 폭발적으로 증가하고 있다. 다양한 빅데이터 처리 플랫폼이 이를 처리하기 위해 등장하고 있다. 하둡(Hadoop)은 이 중 가장 널리 사용되고 있는 플랫폼 중 하나로 Apache 소프트웨어 재단에서 개발했다. 하둡은 빅데이터를 수집, 저장, 처리, 분석, 시각화하는 다양한 서브 프로젝트를 프레임워크로 제공한다. 하둡의 코어 프로젝트인 HDFS(Hadoop Distributed File System)는 블록 기반의 대용량 데이터 저장소로 최소 64MB에서 최대 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 임포트된 데이터의 업데이트나 삭제가 불가능하다. Apache Kudu는 이와 같은 문제를 해결하기 위해 개발되었다. Apache Kudu는 복잡한 질의를

※ 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).
※ 이 논문은 2020년도 한국정보처리학회 춘계학술발표대회에서 'Apache Kudu와 Impala를 활용한 Lambda Architecture 설계'의 제목으로 발표된 논문을 확장한 것임.
† 준 회 원 : 숭실대학교 컴퓨터학과 석사과정
†† 종신회원 : 숭실대학교 컴퓨터학부 교수
Manuscript Received : July 13, 2020
Accepted : August 14, 2020
* Corresponding Author : Yong-Tae Shin(shin@ssu.ac.kr)

처리할 수 없어 주로 Apache Impala와 함께 사용된다.

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

2. 관련 연구

2.1 Lambda Architecture

Lambda Architecture는 오래된 데이터를 보관하는 배치(Batch) 테이블과 실시간 데이터를 가진 실시간 테이블을 조인(Join)하여 결과값을 얻을 수 있도록 구성한 구조이다[2]. 아래 Fig. 1은 Lambda Architecture의 구조를 나타낸다.

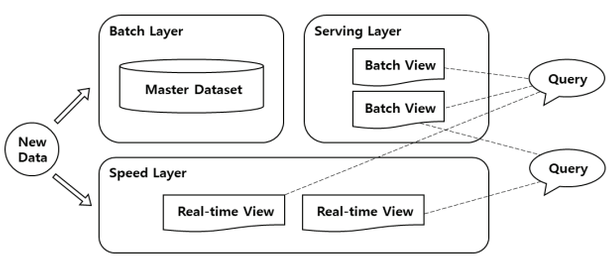

Fig. 1. Structure of Lambda Architecture

Lambda Architecture는 Batch Layer, Speed Layer, Serving Layer로 구성되어 있다. Batch Layer에서는 배치를 이용해 데이터를 미리 계산하여 저장소에 raw 데이터를 보관한다. 배치뷰의 데이터가 부정확할 때 저장소의 raw 데이터를 통해 복구가 가능하다. 기존의 raw 데이터는 새로운 뷰(View)를 제공하고자 할 때 통계 분석이 가능하다. Speed Layer는 데이터를 실시간으로 집계해 별도의 테이블에 저장하여 배치가 실행되는 동안 발생하는 조회에 대한 공백 문제를 해결한다. Serving Layer는 Batch Layer 및 Speed Layer에 저장된 데이터를 조회하기 때문에 빠른 응답이 가능하다.

2.2 Apache Kudu

Apache Kudu는 Apache 소프트웨어 재단에서 개발한 컬럼 지향 데이터 스토리지이다. Apache Kudu는 하둡 기반의 프레임워크 대부분과 호환되며 하둡의 범용 하드웨어 사용성, 확장성, 데이터 가용성 보증의 특성을 지원한다. Apache Kudu는 블록 기반의 스토리지가 아닌 NoSQL OLAP 데이터베이스로 HDFS에서 불가능한 업데이트와 삭제 명령문을 지원한다. Apache Kudu는 데이터를 칼럼 기반으로 저장하여 특정 칼럼만 읽을 때는 디스크에서 읽는 데이터의 양을 줄여 성능을 높인다. Apache Kudu는 일반 DBMS(DataBase Management System)처럼 기본키(Primary Key)를 제공하며, 기본키는 내부적으로 B+트리로 저장되어 대규모 데이터에서 빠르게 원하는 데이터에 접근한다. Apache Kudu는 데이터 저장소 역할 만 하는 플랫폼으로 이를 사용하기 위한 서버가 필요하다[3]. 아래 Fig. 2는 Apache Kudu의 서버 구성을 나타낸다.

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

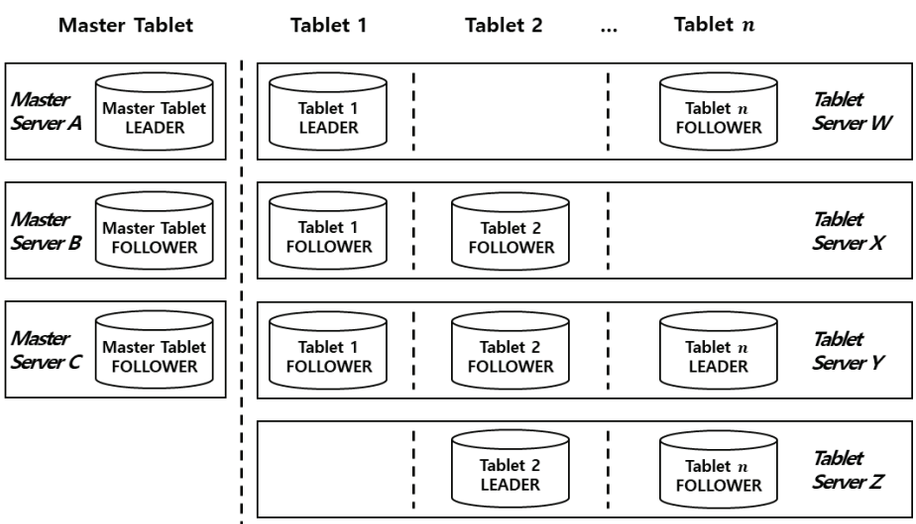

Fig. 2. Server Configuration for Apache Kudu

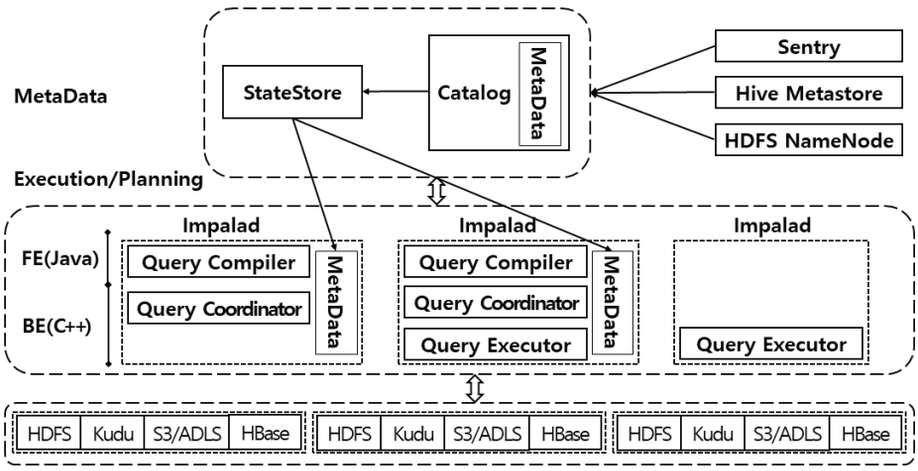

Fig. 3. Configuration of Apache Impala

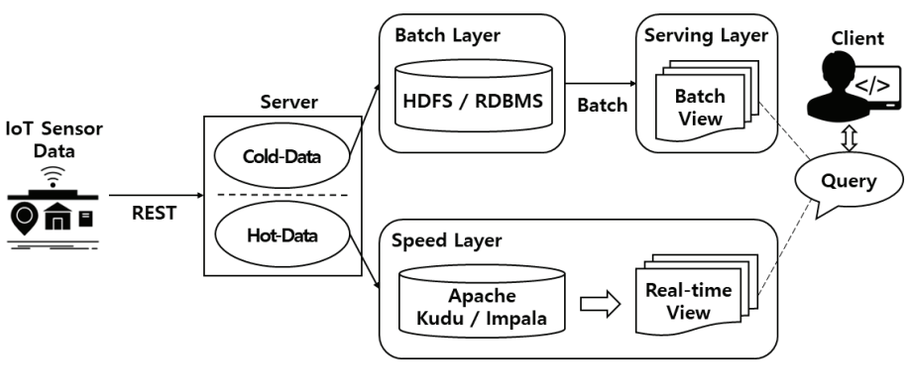

Fig. 4. Suggested Configuration of the Lambda Architecture

2.3 Apache Impala

Apache Impala는 HDFS를 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 맵리듀스(MapReduce)를 이용하지 않는 분산 질의 엔진을 통해 SQL을 실행하여 낮은 지연시간으로 결과를 제공한다. Fig. 3은 Apache 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 하여 Daemon에서 직접 메타데이터를 변경하면 자동으로 동기화된다. Statestore는 Daemon의 상태를 확인하고 메타데이터를 동기화한다.

3. 제안하는 Lambda Architecture 설계

제안하는 Lambda 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 짧으며 크기가 작은 실시간 Hot-Data로 분류한다. 데이터가 크기에 따라 분류되어 각 성격에 맞는 스토리지에 저장됨으로써 HDFS의 Small File 문제를 해결할 수 있다. Speed Layer에서는 실시간 뷰를 끊임없이 생성해 배치가 실행될 때 발생하는 공백 문제를 해결할 수 있다. Fig. 4는 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture의 구성을 나타낸다.

제안하는 구조는 IoT 센서 데이터를 REST 통신으로 서버에 전송하며, 서버는 이를 Cold-Data와 Hot-Data로 분류한다. Cold-Data는 HDFS에 저장되고, HDFS는 배치를 통해 주기적으로 배치 뷰를 생성한다. Hot-Data는 Apache Kudu에 저장되는 동시에 누락된 데이터를 삭제하거나 갱신하여 데이터의 무결성을 보장하며 Impala를 통해 실시간 뷰를 생성한다. 클라이언트는 SQL 질의를 통해 배치 뷰와 실시간 뷰의 조인 결과를 제공받는다.

4. 성능평가

4.1 실험환경 및 평가 방법

제안하는 Lambda Architecture의 성능을 평가하기 위해 하둡 클러스터 환경을 가상환경에 구성한다. 성능평가를 위한 실험환경은 아래 Table 1과 같다.

Table 1. Experimental Environment for Performance Evaluation

	하둡 기반 데이터 수집 환경	제안하는 Lambda Architecture
CPU	Intel(R) Core(TM) i7-9700k 3.60GHz	
RAM	32.0 GB	
HDD	SSD 500 GB	
OS	Linux CentOS 7	
SW	Hadoop 2.9.9 Apache Hbase 2.2.5	Hadoop 2.9.9 Apache Kudu 1.12.0 Apache Impala 3.4.0

성능평가에 사용되는 데이터는 IoT 센서에서 발생하는 데이터를 사용하며, 하둡 기반 데이터 수집기와 제안하는 Lambda Architecture의 성능을 비교하며 평가한다. 아래 Table 2는 성능평가에 사용된 데이터의 명세를 나타낸다.

Table 2. Information of Data

Table Name	Size	Records
sensor_data	2GB	550,000

Columns	Type
device_id	INT
lon	NUMBER
lat	NUMBER
ip	VARCHAR
type	VARCHAR
temp	NUMBER
humi	NUMBER
lux	NUMBER
time	DATETIME
log_info	VARCHAR

성능평가를 위해 위 데이터를 임의의 관계형 데이터베이스에 저장 후 웹 환경에서 1초마다 순차적으로 호출하여 성능평가를 위해 구축한 서버로 데이터를 전송한다.

4.2 기존 하둡 기반 게이터 수집 환경 성능

기존의 하둡 기반 데이터 수집 환경에서는 HDFS에 데이터를 저장하고 하둡의 데이터 스토리지인 Hbase로 배치 뷰를 생성한다. 실험 데이터를 이용해 배치 뷰를 생성할 경우 배치 프로세스의 주기를 최소 5시간에서 최대 20시간으로 설정해야 HDFS에 저장되는 블록의 크기를 최적화할 수 있다. 배치의 실행주기를 5시간 이하 20시간 이상으로 설정할 경우 Small File이 과도하게 생성되어 네임노드에 과부하가 생겨 전체 시스템의 성능이 저하된다.

배치 실행 주기를 T라고 할 때, T만큼의 데이터 조회에 대한 시간 공백이 존재한다. 아래 Fig. 5는 기존 하둡 기반 데이터 수집 환경에서 발생하는 시간 공백을 나타낸다.

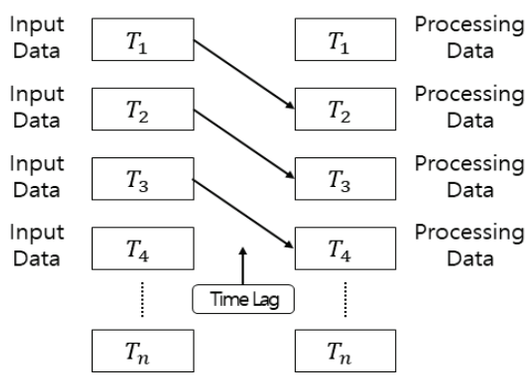

Fig. 5. Time Lag in Batch Process

하둡 기반의 데이터 수집 환경에서 배치 프로세스를 이용하여 결과값에 접근할 경우 시간에 대한 조회 가능한 데이터의 양은 아래 Fig. 6과 같다.

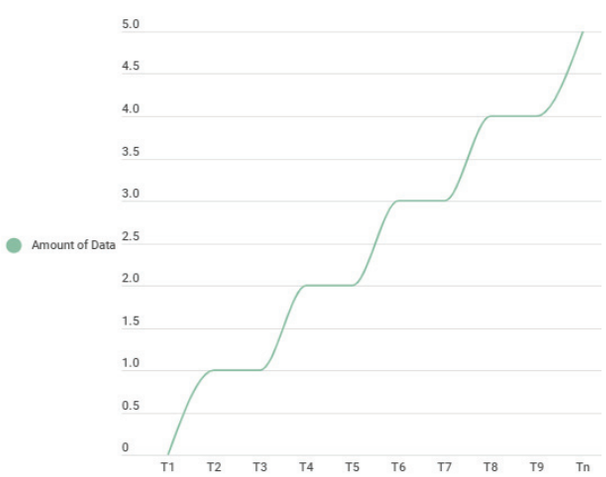

Fig. 6. Amount of Queryable Data by Cycle

4.3 제안하는 Lambda Architecture 데이터 수집 성능

제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 데이터를 Cold-Data와 Hot-Data로 분류하여 실험을 진행한다. 실험 데이터를 IoT 디바이스가 가지고 있

는 메타데이터를 Cold-Data로 정하고 디바이스가 측정하는 Measurement Data를 Hot-Data로 구분한다. Cold-Data는 배치 뷰를 생성하기 위해 HDFS에 저장하며, Hot-Data는 실시간 뷰를 생성하기 위해 Apache Kudu에 저장한다. device_id 칼럼을 배치 뷰와 실시간 뷰를 조인하여 결과를 얻기 위해 Cold-Data에 생성한다. 아래 Fig. 7은 제안하는 Lambda Architecture의 실시간 데이터 처리 구조를 나타낸다.

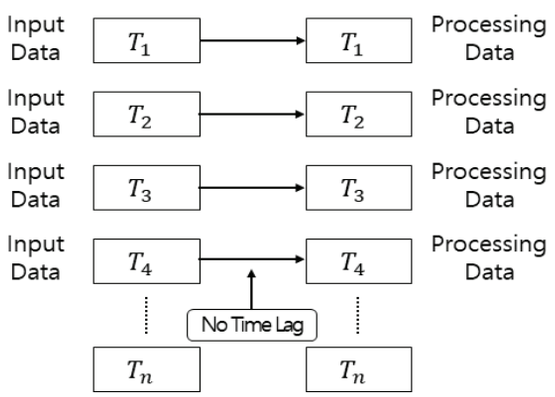

Fig. 7. Real-Time Data Processing Structure

제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 자주 사용되지 않는 Cold-Data를 HDFS에 저장하여 배치 뷰를 생성하고 실시간으로 측정되는 데이터는 Kudu에 저장되어 Impala를 통해 실시간 뷰를 생성한다. 메타데이터는 미리 저장하여 사용하기 때문에 메타데이터로 인해 생기는 데이터의 유실이 발생하지 않는다. 미리 생성된 배치 뷰와 실시간 데이터로 생성되는 실시간 뷰를 사용하여 결과값에 접근하기 때문에 데이터 조회에서 시간으로 인한 공백 문제가 발생하지 않는다. 제안하는 Lambda Architecture의 시간에 따른 조회 가능한 데이터의 양은 아래 Fig. 8과 같다.

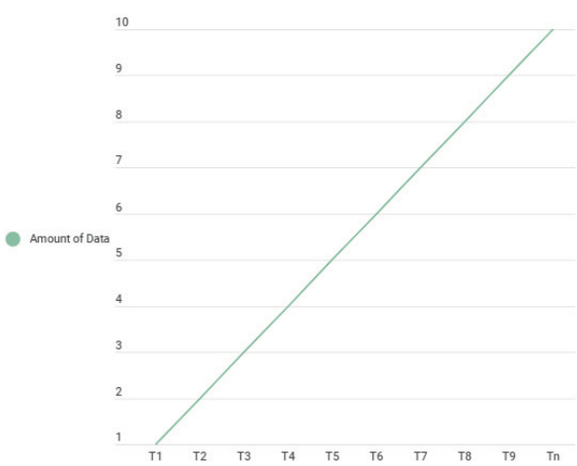

Fig. 8. Amount of Queryable Data over Time

5. 결 론

본 논문에서 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 배치 뷰와 Apache Kudu와 Impala로 생성되는 실시간 뷰를 통해 클라이언트가 결과에 접근할 때 발생할 수 있는 공백 문제를 해결할 수 있다. 또한, 제안하는 Architecture를 사용할 경우, 하둡 기반 데이터 수집 분석 환경에서 발생하는 Small File로 인한 네임노드의 과부하 문제를 해결할 수 있다.

제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture는 실시간으로 결과값을 조회할 수 있어 발생하는 문제를 빠르게 발견하고 조치가 가능하다. 제안하는 구조는 데이터에 대한 지속적인 실시간 모니터링이 가능해 관리 측면에서도 이점을 갖는다. 향후 본 논문에서 제안하는 Apache Kudu와 Impala를 활용한 Lambda Architecture에서 Cold-Data와 Hot-Data를 운영환경에 맞춰 자동으로 분류하는 알고리즘의 연구가 필요하다. 또한, 기존의 Apache Kafka 혹은 Spark를 이용해 구성한 Lambda Architecture 시스템들과의 성능 비교 평가가 필요하다.

References

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

[2] M. Kiran, P. Murphy, I. Monga, J. Dugan, and S. Baveja "Lambda architecture for cost- effective batch and speed big data processing," In: *2015 IEEE International Conference on Big Data (Big Data)*. IEEE, pp.2785-2792, 2015.

[3] T. Lipcon, D. Alves, D. Burkert, J.Cryans, A. Dembo, M. Percy, S. Rus, D. Wang, M. Bertozzi, C. McCabe, and A. Wang "Kudu: Storage for fast analytics on fast data," *Cloudera, inc*, Vol.28, 2015.

[4] M. Kornacker and J. Erickson, "Cloudera impala: Real time queries in apache hadoop, for real," *Ht Tpblog Cloudera Comblog201210cloudera-Impala-Real-Time-Queries—Apache-Hadoop--Real*, 2012.

황 윤 영

https://orcid.org/0000-0002-5239-3796

e-mail : doublewhy@soongsil.ac.kr

2018년 성결대학교 도시계획부동산학부 (학사)

2020년 ~ 현 재 숭실대학교 컴퓨터학과 석사과정

관심분야 : IoT, 빅데이터, 클라우드 컴퓨팅, 네트워크

이 필 원

https://orcid.org/0000-0003-4092-8658
e-mail : pwlee@soongsil.ac.kr
2020년 ~ 현 재 숭실대학교 컴퓨터학과 석사과정
관심분야 : IoT, 클라우드 컴퓨팅, 네트워크

신 용 태

https://orcid.org/0000-0002-1199-1845
e-mail : shin@ssu.ac.kr
1985년 한양대학교 산업공학과(학사)
1990년 Univ. of Iowa, 컴퓨터학과(석사)
1994년 Univ. of Iowa, 컴퓨터학과(박사)
1995년 ~ 현 재 숭실대학교 컴퓨터학부 교수
관심분야 : 정보보호, 인터넷 프로토콜, IoT, 클라우드 컴퓨팅