The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 19, No. 1, pp.117-126, Feb. 28, 2019. pISSN 2289-0238, eISSN 2289-0246

https://doi.org/10.7236/JIIBC.2019.19.1.117
JIIBC 2019-1-16

IoT 환경을 위한 빅데이터 기반 센서 데이터 처리 및 분석

Big Data-based Sensor Data Processing and Analysis for IoT Environment

신동진*, 박지훈**, 김주호***, 곽광진****, 박정민*****, 김정준******

Dong-Jin Shin*, Ji-Hun Park, Ju-Ho Kim***, Kwang-Jin Kwak****, Jeong-Min Park*****, Jeong-Joon Kim******

요 약 IoT 환경에서 발생하는 데이터는 아주 다양하고, 4차 산업혁명의 발전으로 인해 특히 스마트팩토리와 같은 제조 설비 공장에서 발생하는 정형, 비정형 데이터도 확연하게 증가하는 추세이다. 이를 빅데이터 관련 솔루션을 이용하면 다양한 대용량 데이터의 수집, 저장, 처리, 분석 및 시각화 과정을 거쳐 정확한 분석 및 데이터 기반 의사결정을 통한 시스템의 개선 및 확장을 할 수 있다. 따라서 본 논문에서는 IoT 환경에서 사용되는 라즈베리 파이를 이용하여 직접 데이터를 생성하고, 다양한 빅데이터 솔루션을 이용하여 분석한다. 수집에는 Sqoop 솔루션을 이용하여 데이터베이스에서 HDFS로 수집 및 저장하고, 처리에는 Hadoop과 연결되어 병렬 처리가 가능한 Hive 솔루션을 사용하여 데이터를 처리한다. 마지막으로 범용적으로 쓰이는 R 프로그래밍을 통해 처리된 데이터를 분석 및 시각화하여 최종 검증하고자 한다.

Abstract The data generated in the IoT environment is very diverse. Especially, the development of the fourth industrial revolution has made it possible to increase the number of fixed and unstructured data generated in manufacturing facilities such as Smart Factory. With Big Data related solutions, it is possible to collect, store, process, analyze and visualize various large volumes of data quickly and accurately. Therefore, in this paper, we will directly generate data using Raspberry Pi used in IoT environment, and analyze using various Big Data solutions. Collected by using an Sqoop solution collected and stored in the database to the HDFS, and the process is to process the data by using the solutions available Hive parallel processing is associated with Hadoop. Finally, the analysis and visualization of the processed data via the R programming will be used universally to end verification.

Key Words : IoT, Raspberry Pi, Big Data, Data Sensing, Verification

*준회원, 한국산업기술대학교 스마트팩토리융합학과 석사과정
**준회원, 한국산업기술대학교 컴퓨터공학과 학부생
***준회원, 한국산업기술대학교 컴퓨터공학과 학부생
****준회원, 한국산업기술대학교 스마트팩토리융합학과 박사과정
*****정회원, 한국산업기술대학교 컴퓨터공학과 조교수
******정회원, 한국산업기술대학교 컴퓨터공학과 조교수
접수일자 2018년 9월 5일, 수정완료 2019년 1월 2일
게재확정일자 2019년 2월 8일

Received: 5 September, 2018 / Revised: 2 January, 2019 / Accepted: 8 February, 2019
******Corresponding Author: jjkim@kpu.ac.kr
Dept. of Computer Engineering, Korea Polytechnic University, Korea.

Ⅰ. 서 론

제 4차 산업혁명의 발달로 인한 IoT 환경에서 발생하는 데이터는 다양하고 많은 양을 지니고 있다. 공장 내부를 모니터링할 수 있기 때문에 공정의 생산과정을 예측하여 공정을 효율적으로 운영할 수 있다. 또한, 제조 설비 공장에서 온도, 습도 같은 데이터는 공장의 생산 과정에서 발생할 수 있는 예외 상황을 판단하기에 좋은 데이터가 된다[1].

IoT 장비는 센서를 통해 사물과 무선 통신을 이용하여 연결되기 때문에 효율성이 좋은 장비이다. 또한, 센서의 크기도 작고, 본 연구에서 사용하는 라즈베리 파이는 크기에 비해 성능이 우수하므로 4차 산업혁명에서 IoT를 이용한 환경은 필수적인 요소라 할 수 있다[2].

본 논문에서는 라즈베리 파이를 이용하여 온도, 습도 데이터를 생성하고, 데이터베이스로 전송하여 저장되면, 다양한 빅데이터 솔루션을 사용하여 데이터를 수집, 저장, 처리, 분석 및 시각화하는 연구를 진행하였으며, 구성은 다음과 같다. 2장에서는 본 연구에 수행되는 빅데이터 관련 솔루션을 하고, 3장에서는 본 연구에서 제시하는 시스템의 구조와 상세 프로세스를 설명한다. 마지막으로 4장에서는 결론과 향후 연구 내용에 대하여 설명한다.

Ⅱ. 관련 솔루션

1. 빅데이터(Big Data)

4차 산업혁명에 주 기술인 빅데이터는 기존에 사용하던 테이블 형태의 정형 데이터를 저장하는 RDBMS에 비해 XML, JSON 등 다양한 형식의 데이터 구조를 분산 저장할 수 있는 기술이다. 또한, 빅데이터 처리 과정은 주로 수집, 저장, 처리, 분석 및 시각화 과정을 거쳐 대용량의 데이터를 편리하고, 빠르게 분석할 수 있기 때문에 현 시점에서 필수적인 기술이라 할 수 있다[3].

2. 라즈베리 파이(Raspberry Pi)

라즈베리 파이는 현재 IoT 환경에서 가장 널리 쓰이고, 다양한 센서 데이터를 생성할 수 있다. 또한, 작은 크기임에도 불구하고, 성능이 우수한 CPU, GPU와 HDMI 연결을 통한 고해상도의 HD Video를 지원할 뿐 아니라, 일반 데스크톱에서 사용하는 주변기기인 키보드, 마우스 등을 사용할 수 있기 때문에 편리하다[4].

온도, 습도, 초음파, 동작 감지, 적외선 등 다양한 센서를 지원하며, 센서 데이터를 생성할 때 사용하는 프로그래밍 언어도 C언어, Java, Python 등 다양하게 지원하기 때문에 범용성이 큰 IoT 장비라 할 수 있다.

3. 스쿱(Sqoop)

스쿱은 정형 데이터를 저장하는 RDBMS에서 빅데이터의 저장 솔루션인 하둡으로 분산 저장할 수 있는 수집 솔루션이다. 지원하는 데이터베이스 종류는 Mysql, Oracle 등 다양하게 사용 가능하며, 하둡에서 사용되는 Map-Reduce 작업을 통해 수집된다. 데이터베이스에서 빅데이터 분산 저장 시스템 하둡으로 수집하는 과정은 Import 과정이라 불리며, 반대로 하둡에서 RDBMS로 데이터를 전송하는 과정은 Export 과정으로 분류된다.

4. 하둡(Hadoop)

하둡은 빅데이터 저장 솔루션에서 가장 많이 사용되는 기술이다. 하둡은 크게 대용량의 데이터를 분산 저장하는 HDFS(Hadoop Distributed File System)와 Key-Value 구조를 이용한 처리 방법인 Map-Reduce 두 가지로 구분된다[5].

HDFS의 경우 파일을 분산 저장할 때 2.9 버전까지는 3개의 복제본을 이용한 Replication 기법을 사용했으며, 3.0버전부터는 Parity Bit를 이용한 Erasure Coding 기법을 사용하고 있다[6].

Map-Reduce의 경우 일반적인 처리에 비해서 속도가 빠르지만, Java 코딩과 Key-Value 구조를 이용한 방법은 접근하기에 어려움이 있어 최근 Hive, Pig 등 다른 솔루션을 이용하여 Map-Reduce 형식으로 처리한다[7].

5. 하이브(Hive)

하이브는 빅데이터 처리에서 사용되는 병렬 처리 솔루션으로 Map-Reduce 형식의 구조를 사용하기에 어려운 사용자들을 위해 제공된 솔루션이다. HDFS와 연동이 되기 때문에 HDFS에 데이터가 저장되어있으면 하이브에서 편리하게 데이터를 불러와 SQL 형식의 스키마 구조를 통해 데이터를 관리하고, 처리할 수 있다. 이를 HiveQL이라고 불리며, 하이브에서 사용되는 HiveQL의 경우 RDBMS에서 사용하는 SQL과 문법이 유사하기 때문에 편리함을 제공한다[8].

Ⅲ. 시스템 구조 및 상세 프로세스

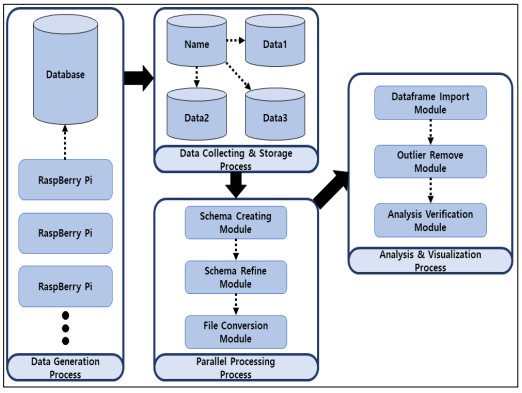

그림 1. 빅데이터 기반 센서 데이터 처리 및 분석 시스템 구조도
Fig. 1. System Architecture of Big Data-based for Sensor Data Processing and Analysis

빅데이터 기반 센서 데이터 처리 및 분석 시스템은 그림 1과 같이 '데이터 생성 프로세스', '데이터 수집 및 저장 프로세스', '병렬 처리 프로세스', '분석 및 시각화 프로세스' 4가지로 설계 및 구현된다.

'데이터 생성 프로세스'에서 사용한 하드웨어는 라즈베리 파이 B+ 버전이고, 데이터 생성을 위해 사용한 센서는 DHT11을 사용하였다. DHT11 센서는 온도, 습도 데이터를 생성해주며, 생성되는 데이터는 자동으로 MariaDB 서버로 전송된다. 데이터베이스는 네이버에서 제공하는 MaridDB 서버를 이용하였다.

'데이터 수집 및 저장 프로세스'에서 데이터 수집에는 빅데이터 수집 솔루션의 'Sqoop'을 이용하여 데이터베이스의 테이블형태 데이터를 Hadoop의 'HDFS'로 저장한다. 데이터 저장에는 4개의 노드를 통한 완전분산모드 구성을 통하여 수집되는 데이터를 분산하여 저장한다.

'병렬 처리 프로세스'에서는 Hadoop 프레임워크 상위에서 동작하는 빅데이터 처리 솔루션의 'Hive'를 이용하여 데이터 처리 및 정제가 가능하다.

'분석 및 시각화 프로세스' 에서는 처리가 완료된 데이터를 빅데이터 분석 및 시각화 솔루션의 R 프로그래밍을 이용하여 분석하고, 본 본문에서 구현한 시스템의 과정이 완료된다.

1. 데이터 생성 프로세스

데이터 생성 프로세스는 IoT 장비 중 하나인 라즈베리 파이를 이용하여 데이터를 생성하며, 사용하는 센서는 온도, 습도 센서인 DHT11을 사용한다.

생성되는 온도, 습도 데이터는 생성이 될 때마다 자동으로 Mysql로 저장되며, 저장되는 값은 온도, 습도 데이터를 포함한 데이터 발생 시간 값까지 저장한다.

데이터 생성 프로세스는 라즈베리 파이에서 온도, 습도 센서인 DHT11을 이용하였으며, 센서에 대한 구조와 Mysql의 스키마 구조, 마지막으로 데이터 생성에 사용한 Python 코드를 설명한다.

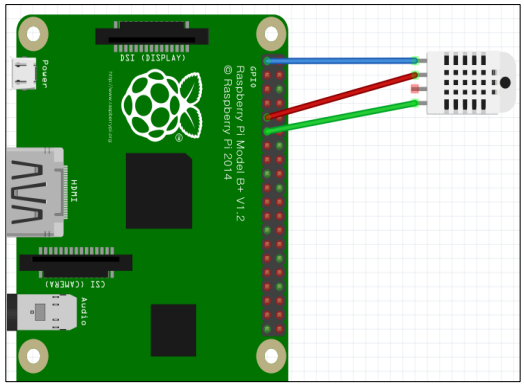

그림 2. 라즈베리 파이와 DHT11 센서 연결 구조
Fig. 2. Raspberry Pi and DHT11 sensor connection structure

DHT11 온도, 습도 센서와 라즈베리 파이를 연결한 구조는 그림 2와 같다. 파란색 연결부는 센서의 (+)극을 담당하는 전원, 초록색 연결부는 센서의 (-)극을 담당하는 전원이다. 빨간색은 Python 코드를 통해서 센서를 제어할 때 사용하는 연결부가 된다.

```
MariaDB [DHT]> show tables;
+--------------+
| Tables_in_DHT |
+--------------+
| collect_data  |
+--------------+
1 row in set (0.00 sec)

MariaDB [DHT]> desc collect_data;
+-------+----------+------+-----+---------+-------+
| Field | Type     | Null | Key | Default | Extra |
+-------+----------+------+-----+---------+-------+
| time  | datetime | NO   |     | NULL    |       |
| tem   | float    | YES  |     | NULL    |       |
| hum   | float    | YES  |     | NULL    |       |
+-------+----------+------+-----+---------+-------+
3 rows in set (0.01 sec)
```

그림 3. 센서 데이터가 저장되는 스키마 구조
Fig. 3. Schema structure in which sensor data is stored

그림 3은 Python 코드에서 생성되는 센서 데이터를 전송하여 저장되는 Mysql의 스키마 구조를 나타낸다. 데이터가 발생되는 시간, 온도 값, 습도 값 3가지 필드로 구성되어 있으며, 시간은 datetime 형식을 이용하고, 온도와 습도는 숫자 값으로 수집된다.

```
1  import RPi.GPIO as GPIO
   import Adafruit_DHT as dht
   import pymysql
   import time
   import sys
2
   db = pymysql.connect(host="localhost", user="root",
   password="bigdata", db="DHT")
3  try:
    with db.cursor() as cur:
4    sql="insert into collect_data values(%s, %s, %s)"
5    while True:
      h, t = dht.read_retry(dht.DHT11, 17)
      if h is not None and t is not None:
6      print(time.strftime("%Y-%m-%d %H:%M:%S",
   time.localtime()), 'Temp=%0.1fC Humidity=%0.1f'%(t, h)
7      cur.execute(sql, (time.strftime("%Y-%m-%d
   %H:%M:%S", time.localtime()), t, h))
8      db.commit()
      else:
       print("Failed to get reading.")
      time.sleep(1)
   except KeyboardInterrupt:
    exit()
   finally:
    db.close()
```

그림 4. Python을 이용한 온도, 습도 데이터 생성
Fig. 4. Generate temperature and humidity data using Python

그림 4는 Python을 사용하여 온도, 습도 센서의 데이터를 발생시켜, Mysql로 전송하여 저장하는 코드이다.

1) 센서를 Python 코드의 BCM 번호를 적용하기 위해 GPIO, 온도, 습도를 제어하는 Adafruit_DHT, 데이터를 Mysql로 전송하는 pymysql, 센서가 측정되는 시간 값을 저장하기 위한 time, 모듈의 저장된 위치를 불러오는 sys 라이브러리를 코드에 적용한다. 2) Mysql로 접속하기 위한 호스트주소, 유저명, 비밀번호, 사용할 데이터베이스 이름을 입력한다. 3) 발생하는 센서 데이터를 Mysql에 접속 후 테이블에 입력해주는 SQL 쿼리문이다.

4) 온도, 습도 제어 라이브러리 Adafruit_DHT를 이용하여 온도, 습도 데이터를 생성한다. 5) 시간, 온도, 습도 데이터가 발생하면 라즈베리 파이 콘솔에 생성되는 데이터를 출력한다. 6) Mysql에 접속하여 생성된 시간, 온도, 습도 데이터를 3)의 데이터 입력 insert 쿼리문을 통해 데이터를 저장한다.

7) 저장이 완료되면 데이터베이스를 commit하고, 예외적으로 센서 데이터가 생성되지 않으면 "Failed to get reading"이라는 메시지와 함께 오류를 출력한다. 8) 센서 생성이 완료되면, 키보드 인터럽트인 Ctrl + C 단축키를 이용하여 종료할 수 있으며, 최종적으로 데이터베이스의 접속을 종료하게 된다.

```
MariaDB [DHT]> select * from collect_data limit 10;
+---------------------+------+------+
| time                | tem  | hum  |
+---------------------+------+------+
| 2018-08-22 02:42:11 |   47 |   49 |
| 2018-08-22 02:42:13 |   46 |   44 |
| 2018-08-22 02:42:14 |   46 |   52 |
| 2018-08-22 02:42:16 |   46 |   52 |
| 2018-08-22 02:42:17 |   46 |   52 |
| 2018-08-22 02:42:19 |   46 |   52 |
| 2018-08-22 02:42:20 |   46 |   52 |
| 2018-08-22 02:42:22 |   46 |   52 |
| 2018-08-22 02:42:23 |   46 |   52 |
| 2018-08-22 02:42:25 |   46 |   52 |
+---------------------+------+------+
10 rows in set (0.00 sec)
```

그림 5. 테이블에 저장된 센서 데이터 확인
Fig. 5. Check the sensor data stored in the table

그림 5는 Mysql에서 생성한 collect_data 테이블에 생성된 시간, 온도, 습도 데이터가 입력되었는지 확인하는 select 쿼리문이다.

time 필드에는 연도-월-일 시간-분-초 형식으로 데이터가 저장되고, tem 필드와 hum 필드에는 온도 값, 습도 값이 숫자 형식으로 저장되었다.

Mysql에 데이터가 저장되면 이후 데이터 수집 및 저장 프로세스에서 Sqoop 솔루션을 사용하여 수집할 때 Sqoop에서는 기본 키 필드가 존재하여야 수집이 되기 때문에 기본 키 추가작업을 그림 6처럼 실행한다.

```
mysql> alter table collect_data add column id int(20) unsigned primary KEY AUTO_INCREMENT;
Query OK, 0 rows affected (0.44 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc collect_data;
+-------+-----------------+------+-----+---------+----------------+
| Field | Type            | Null | Key | Default | Extra          |
+-------+-----------------+------+-----+---------+----------------+
| time  | datetime        | YES  |     | NULL    |                |
| tem   | float           | YES  |     | NULL    |                |
| hum   | float           | YES  |     | NULL    |                |
| id    | int(20) unsigned | NO   | PRI | NULL    | auto_increment |
+-------+-----------------+------+-----+---------+----------------+
4 rows in set (0.00 sec)

mysql> select * from collect_data limit 3;
+---------------------+------+------+----+
| time                | tem  | hum  | id |
+---------------------+------+------+----+
| 2018-08-22 05:28:34 |   26 |   44 |  1 |
| 2018-08-22 05:28:35 |   26 |   39 |  2 |
| 2018-08-22 05:28:36 |   26 |   50 |  3 |
+---------------------+------+------+----+
```

그림 6. 스키마 구조에 기본 키 필드 추가
Fig. 6. Add a primary key field to the schema structure

alter table 쿼리문을 통해서 필드명이 id이며, 첫째 튜플부터 1을 시작하여 증가하는 옵션을 추가하고, id 필드를 기본 키로 설정하는 추가작업을 한다.

쿼리문이 완료되고, 스키마 구조를 확인하면 id 필드가 추가되며, select 쿼리문을 통해서 입력된 값을 확인하면 정상적으로 필드 추가가 이루어졌다.

2. 데이터 수집 및 저장 프로세스

데이터 수집 및 저장 프로세스는 수집 솔루션 중 스쿱을 이용해서 Mysql에 저장되어있는 시간, 온도, 습도 데이터를 하둡의 HDFS로 분산 저장한다.

HDFS에 저장되는 데이터는 “,” 콤마 형식으로 구분되어 저장되며, 저장되는 데이터는 “2018-08-01 12:33:40, 45, 50” 형식으로 저장되며, 사용하는 map 개수에 따라 part-m-00000, part-m-00001과 같은 파일 이름에 숫자 값을 늘려가며 저장된다.

```
sudo ./sqoop import --connect
jdbc:mysql://106.10.53.150:3306/DHT --table collect_data
--username root --password bigdata --target-dir /Sensor
```

그림 7. 스쿱을 이용한 센서 데이터 수집 명령어
Fig. 7. Command to collect sensor data using Sqoop

그림 7은 4개의 노드 중 네임노드에서 Mysql에 저장되어있는 센서 데이터를 수집하는 명령어다. 라즈베리파이가 데이터를 전송하여 데이터가 저장되어있는 Mysql의 IP 106.10.53.150, Port 3306, 데이터베이스 이름 DHT, 테이블 명 collect_data, Mysql에 접속하는 root 유저, 비밀번호 bigdata, 수집하는 센서 데이터를 HDFS의 분산 저장되는 위치와 디렉토리명을 입력해서 명령어를 수행하면, 그림 8과 같이 데이터가 저장된다.

```
hadoop@hadoop-name:~/mysql$ hadoop fs -ls /Sensor/
18/08/22 17:27:28 WARN util.NativeCodeLoader: Unable to load native-hadoop libra
ry for your platform... using builtin-java classes where applicable
Found 5 items
-rw-r--r--   3 root supergroup          0 2018-08-22 17:27 /Sensor/_SUCCESS
-rw-r--r--   3 root supergroup      33912 2018-08-22 17:27 /Sensor/part-m-00000
-rw-r--r--   3 root supergroup      34911 2018-08-22 17:27 /Sensor/part-m-00001
-rw-r--r--   3 root supergroup      34965 2018-08-22 17:27 /Sensor/part-m-00002
-rw-r--r--   3 root supergroup      35002 2018-08-22 17:27 /Sensor/part-m-00003
```

그림 8. 수집 완료된 데이터 목록 확인
Fig. 8. Check the list of collected data

스쿱 명령어에서 지정한 디렉토리를 hadoop fs -ls 명령어를 통해 조회하면 정상적으로 데이터가 수집된 것을 그림 8에서 알 수 있다.

```
hadoop@hadoop-name:~/mysql$ hadoop fs -cat /Sensor/part-m-00000
18/08/22 17:39:00 WARN util.NativeCodeLoader: Unable to load nat
ry for your platform... using builtin-java classes where applica
2018-08-22 05:28:34.0,26.0,44.0,1
2018-08-22 05:28:35.0,26.0,39.0,2
2018-08-22 05:28:36.0,26.0,50.0,3
2018-08-22 05:28:37.0,26.0,52.0,4
2018-08-22 05:28:38.0,26.0,50.0,5
2018-08-22 05:28:39.0,26.0,53.0,6
2018-08-22 05:28:40.0,26.0,50.0,7
2018-08-22 05:28:41.0,26.0,66.0,8
```

그림 9. 수집 완료된 데이터 내용 확인
Fig. 9. Check the contents of collected data

데이터를 hadoop fs -cat 명령어로 확인하면 Mysql에서 저장되었던 데이터 형식 및 구조가 같은 것을 그림 9에서 볼 수 있다.

3. 병렬 처리 프로세스

병렬 처리 프로세스는 처리 솔루션 중 하이브를 이용하며, 기존에 구성되어 있는 4개의 노드를 이용해서 병렬처리하게 된다.

하이브에서 ‘Schema Creating Module’을 통해 스키마를 생성하여 데이터를 읽어 들이고, ‘Schema Refine Module’을 통해 분석에 불필요한 데이터를 정제하며, ‘File Conversion Module’을 통해 분석에 필요한 파일로 변환되어 처리가 완료된다.

가. Schema Creating Module

```
CREATE TABLE sensor_data(
time timestamp,
tem int,
hum int,
id int)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE;
```

그림 10. 하이브 스키마 구조 생성
Fig. 10. Create Hive schema structure

수집된 데이터를 하이브에서 스키마 구조를 생성하여 데이터를 입력하기 위해선 테이블을 생성하여야 한다. 필드의 데이터 타입은 시간을 저장하는 형식은 timestamp, 온도와 습도는 소수점으로 구분하여 수집하였지만, 실제 데이터에서 소수점은 전부 0으로 표현되기 때문에 int 형식으로 생성하고, 마지막으로 스쿱에서 수집하기 위해 생성했던 기본 키의 id 필드도 int 형식으로 생성한다. 마

지막으로 필드를 구분하는 구분자는 “,” 콤마를 기준으로 주고, 데이터의 레코드 구분은 “\n” 엔터를 기준으로 설정하여 테이블 생성을 완료하였다.

나. Schema Refine Module

스키마 구조 생성이 완료되면 데이터를 입력하고, 분석에 필요한 필드만 유지한 채 나머지 필드를 제거하는 처리 작업이 필요하다.

```
1  LOAD DATA INPATH "/Sensor/*"
   INTO TABLE sensor_data;

2  CREATE TABLE sensor_refine
   AS SELECT time, tem, hum from sensor_data;
```

그림 11. 하이브 스키마 처리 명령어
Fig. 11. Hive schema processing instruction

1) 스쿱을 이용해 수집된 센서 데이터들의 경로를 입력해주고, 모든 센서 데이터를 sensor_data 테이블로 입력한다. 입력이 완료된 후 데이터를 확인하면 시간, 온도 값, 습도 값, id 값으로 저장이 된다. 2) id 값은 분석에 불필요하기 때문에 처리 과정을 거친다. 기존에 저장된 sensor_data 테이블에서 time, tem, hum 필드만 sensor_refine 테이블로 저장한다.

다. File Conversion Module

처리가 완료된 테이블의 데이터는 id를 제외한 시간, 온도, 습도 값만 유지하고 있다. 이 데이터를 분석 및 시각화 프로세스에서 사용하기 위해서는 파일을 읽을 수 있는 구조로 변환해주어야 한다.

```
1  INSERT OVERWRITE LOCAL DIRECTORY
   '/home/hadoop/sensor_refine' row format delimited
   fields terminated by ',' select * from sensor_refine;

2  cd ~/sensor_refine && mv 000000_0 sensor_refine.csv
```

그림 12. 분석 가능한 파일 타입 변환 과정
Fig. 12. Analyzeable file type conversion process

1) 일반적으로 분석에서는 csv 타입의 “,” 콤마로 필드가 구분되는 파일 타입을 주로 쓰기 때문에 sensor_refine 테이블의 모든 데이터를 콤마를 기준으로 리눅스 로컬의 /home/hadoop/sensor_refine 디렉토리로 저장한다. 2) 리눅스 로컬의 홈디렉토리 아래 sensor_refine 디렉토리로 이동하고, 파일 이름을 sensor_refine.csv 로 변경하여 분석에 사용 가능한 타입으로 변경한다.

```
hadoop@hadoop-name:~/sensor_refine$ cat sensor_refine.csv
2018-08-22 05:28:34,26,44
2018-08-22 05:28:35,26,39
2018-08-22 05:28:36,26,50
2018-08-22 05:28:37,26,52
2018-08-22 05:28:38,26,50
2018-08-22 05:28:39,26,53
2018-08-22 05:28:40,26,50
```

그림 13. 최종 처리가 완료된 데이터
Fig. 13. Final processed data

마지막 과정까지 처리가 완료된 데이터의 구조는 그림 13과 같다. 첫 레코드를 살펴보면 첫 번째 필드인 “2018-08-22 05:28:34”는 온도, 습도 데이터가 발생했을 때의 시간, 두 번째 필드인 “26”은 온도 값, 세 번째 필드인 “44”는 습도 값이다.

처리가 끝난 데이터는 R 프로그래밍을 이용해서 분석 및 시각화 프로세스에서 분석 및 시각화를 진행한다.

4. 분석 및 시각화 프로세스

분석 및 시각화 프로세스는 R 프로그래밍을 이용하고, 사용하는 분석기법은 회귀기법을 이용하여 온도와 습도의 관계에 대해서 분석한다.

‘Dataframe Import Module’을 통해 분석 파일을 데이터 프레임 구조로 입력하고, ‘Outlier Remove Module’을 통해 비정상적으로 높거나 낮은 값인 이상치를 제거하며, ‘Analysis Verification Module’을 통해 회귀모델을 적용하여 온도와 습도의 관계를 분석한다.

가. Dataframe Import Module

```
1  x<-read.csv("sensor_refine.csv",col.names=c("Time"
   ,"Temperature","Humidity"))

   x$Time <- as.POSIXct(x$Time,
2  format="%Y-%m-%d %H:%M:%S")

   ggplot(data=x, aes(x=x$Time, y=x$Temperature)) +
     geom_line(color="red", size=1) + xlab("시간") +
3  ylab("온도(℃)와 습도(%)") +
     geom_line(aes(x=x$Time, y=x$Humidity),
   color="blue") +
     ggtitle("시간에 따른 온도 그래프") +
   theme(plot.title = element_text(hjust = 0.5))
```

그림 14. 시간에 따른 온도, 습도 그래프 코드
Fig. 14. Temperature and humidity graph code

1) x 변수에 csv 형식의 파일을 읽고, 필드명을 “Time”, “Temperature”, “Humidity”로 지정하여 데이터 프레임 구조를 만든다. 2) 데이터를 읽어 들이면 “Time” 필드의 값은 시간이지만 형태는 Factor 구조를 지니고 있기 때문에 as.POSIXct 함수를 통해 시간으로 형태를 바꾸어 준다.

3) x 변수에 저장된 값을 ggplot 라이브러리를 이용해서 그래프를 그려준다. 그래프의 가로는 시간의 변화, 세로에서 온도는 빨간색, 습도는 파란색으로 설정하여 그리면 그림 15와 같이 시각화된다.

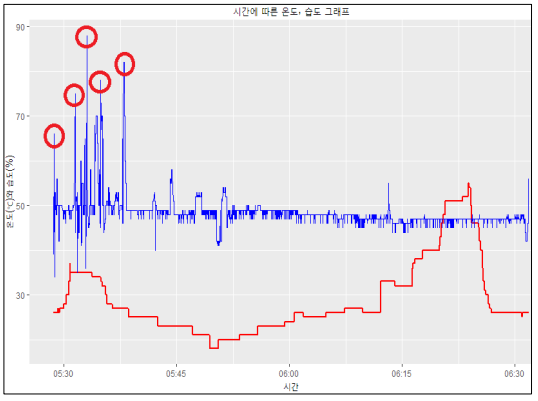

그림 15. 시간에 따른 온도, 습도 그래프 시각화
Fig. 15. Visualization of temperature and humidity graph overtime

시간에 따른 온도, 습도 그래프를 확인하면 온도와 습도와의 관계에 대해서 정확하게 분석이 어려우며, 센서의 불안정한 동작 때문에 발생한 이상치를 볼 수 있다.

R 프로그램에 내장되어있는 Boxplot 함수를 통하여 이상치 데이터값을 확인하고, 제거하는 과정을 거쳐 최종 분석을 한다.

나. Outlier Remove Module

```
1  x <- subset(x, select = -c(Time))
2  pairs.panels(x)
```

그림 16. 시간 값 제거 및 상관관계 분석
Fig. 16. Time value elimination and correlation

1) 온도와 습도의 관계를 확인하기 위해 subset 함수를 사용하여 Time 필드의 시간 값을 제거한다. 2) 온도와 습도의 상관관계를 확인하기 위해서 pairs 라이브러리의 panels 함수를 이용하여 시각화한다.

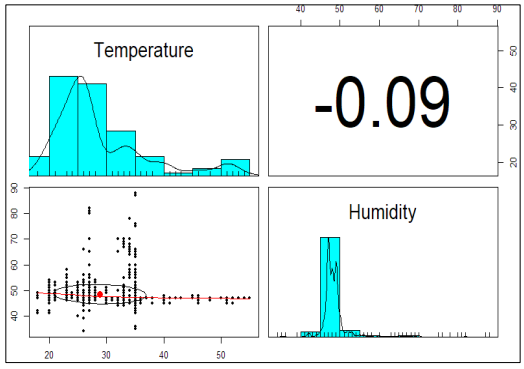

그림 17. 온도와 습도의 상관관계 분석
Fig. 17. Analysis of correlation between temperature and humidity

그림 17은 온도와 습도의 상관관계를 시각화로 표현하였다. 상단의 -0.09 수치의 마이너스는 반비례를 의미하며, 이 수치는 1에 가까울수록 비례 관계, -1에 가까울수록 반비례 관계를 나타낸다. 또한, 하단의 산점도에서 동그라미 도형으로 표현된 부분은 -0.09 수치와 관련된 데이터의 밀집도를 의미한다.

산점도를 확인하면 불규칙적으로 높은 이상치 값을 확인할 수 있으며, -0.09의 수치는 관련이 거의 없다고 볼 수 있다.

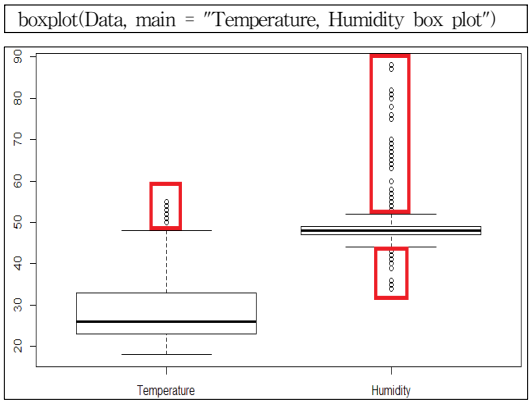

그림 18. boxplot 함수를 통한 이상치 확인
Fig. 18. Outlier check with boxplot function

R 프로그램에 내장된 이상치를 확인할 수 있는 boxplot 함수를 이용하여 이상치 확인 그래프를 그린다.

그림 18에서 빨간색 네모박스로 표시된 데이터는 이상치로 판단되며, 값의 수치는 온도는 48 값 이상, 습도는 52 값 이상 및 44 값 이하로 보인다.

```
1  Data <- subset(x, Temperature < 48 & Humidity < 52)
2  boxplot(Data, main = "Temperature, Humidity box plot")
```

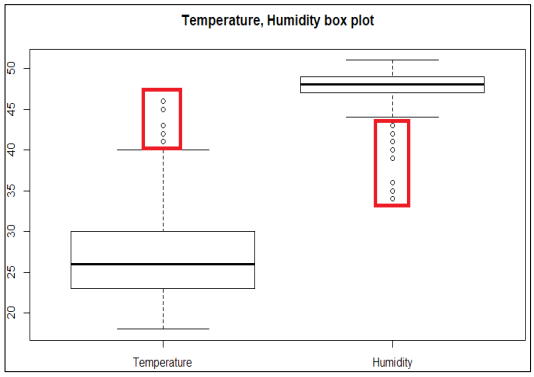

그림 19. 이상치 제거 후 이상치 재확인
Fig. 19. Reconcile the outliers after removing the outliers

1) subset 함수를 이용하여 온도는 48 값 이하만, 습도는 52 이하의 값만 유지한 채 나머지 값을 삭제한다.

2) 삭제 완료 후 boxplot 함수로 다시 이상치를 확인하면 그림 18에서 나온 결과와 다른 결과를 볼 수 있으며, 이상치가 줄어든 것을 확인할 수 있다.

```
1  Data <- subset(Data, Temperature < 40 & Humidity > 44)
2  ...
3  pairs.panels(Data)
```

그림 20. 이상치 반복 제거 후 상관관계 분석
Fig. 20. Analysis of correlation after outlier removal

1) 그림 19에서 확인된 온도는 40 값 이하, 습도는 44 값 이상의 데이터만 유지한 채 나머지는 삭제한다. 2) 온도와 습도의 상관관계를 정확하게 확인하고, 상관관계에서 반비례 관계의 수치를 높여주기 위해 이상치 제거 과정을 반복한다.

3) 이상치 제거가 완료되면, 온도와 습도의 상관관계를 재확인하기 위해서 pairs 라이브러리의 panels 함수를 이용하여 시각화한다.

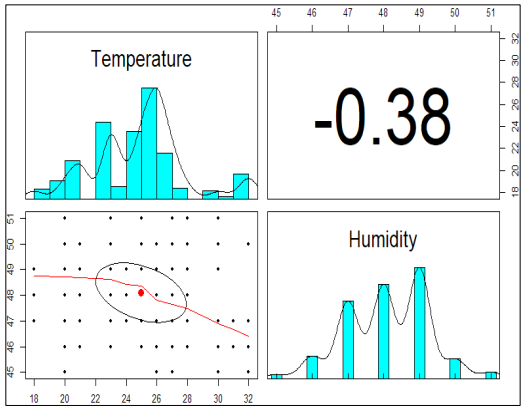

그림 21. 온도와 습도의 상관관계 재확인
Fig. 21. Correlation reaffirmation of the temperature and humidity

기존의 그림 17의 -0.09 수치에 비해서 더 높은 상관관계 수치인 -0.38을 확인할 수 있으며, 하단의 산점도를 확인하면 밀집된 데이터가 반비례 관계를 지닌 것을 볼 수 있다.

즉, 온도가 높아질수록 습도는 내려가는 반비례 관계가 더욱더 정확하게 분석된 것을 확인할 수 있다.

다. Analysis Verification Module

```
ggplot(data = Data, aes(x=Temperature, y=Humidity))+geom_count()+geom_smooth(method="lm")
```

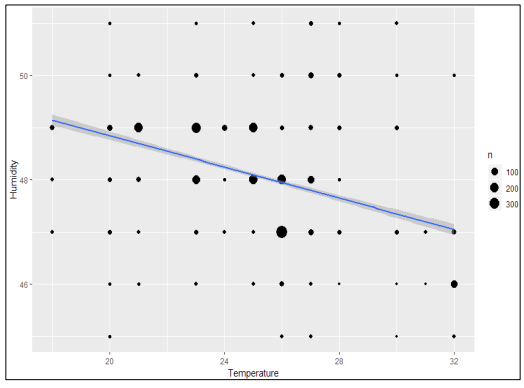

그림 22. 회귀모델을 적용한 온도, 습도 관계 그래프
Fig. 22. Graph of temperature and humidity with regression model applied

그림 22는 ggplot 라이브러리를 이용해 이상치 제거가 완료된 데이터를 입력하고, 가로에 온도, 세로에 습도를 표시하여 관계를 나타낸 그래프이다. 그래프의 동그라미 부분은 데이터의 산점도를 나타내며, 동그라미의 크기가

커질수록 위치한 데이터의 양이 많은 것을 의미한다.

geom_smooth(method="lm") 함수를 이용하여 회귀모델을 적용하여 온도와 습도의 상관관계를 선으로 표현하였다. 확인해본 결과 반비례 영향을 가장 많이 받는 데이터는 온도는 20도에서 26도, 습도는 46%에서 49%로 보인다.

Ⅳ. 결론

본 논문에서는 IoT 환경에서 자주 사용되는 라즈베리 파이를 이용하여 데이터를 생성하며, 생성되는 데이터를 데이터베이스로 전송하고, 빅데이터 처리 과정을 통해 정제된 온도와 습도 데이터의 관계를 검증하는 연구를 수행하였다. 라즈베리 파이에서 발생하는 데이터는 작은 값이지만, 분석을 위해 오랫동안 데이터를 생성하면 대용량의 데이터가 생성되며, 수집, 저장, 처리, 분석에 사용되는 빅데이터의 주요 솔루션 활용 용도가 증가하게 된다.

일반적으로 온도와 습도의 관계를 분석하면 본 논문에서 사용한 온도와 습도를 제외하고, 기후, 기압, 밀도, 수증기량 등 다양한 요인을 가지고 있는 데이터가 필요하다. 하지만, 라즈베리 파이에서 제공되는 센서 종류는 한정되어 있으며, 라즈베리 파이를 통한 데이터의 수집보다 다양한 요인을 분석하기 위해서는 추가적인 하드웨어나 소프트웨어의 기술이 필요하다. 따라서, 온도와 습도에 영향을 주는 요인으로 판단되는 기압, 밀도 등 다양한 센서를 활용하여 향후 날씨까지 예측할 수 있는 연구를 수행할 예정이다.

References

[1] Hyoung-Ro Lee, Cho-Ho Lin, "Design and Implementation of Smart Home Security Monitoring System based on Raspberry Pi2," Journal of The Institute of Internet, Broadcasting and Communication, Vol. 16, No. 5, pp. 131-136, Oct, 2016.
DOI: https://doi.org/10.7236/JIIBC.2016.16.5.131

[2] Jayavardhana Gubbi, Rajkumar Buyyab, Slaven Marusica, Marimuthu Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Future generation computer systems, Vol. 29, No. 7, pp. 1645-1660, Feb, 2013.
DOI: https://doi.org/10.1016/j.future.2013.01.010

[3] Dong-Jin Shin, Ho-Geun Lee, Jong-Min Eun, Jeong-Joon Kim, Jeong-Min Park, "Bigdata-based Anomaly Detection Technology in Web Services Environment," Journal of Asia-pacific Multimedia Services Convergent with Art, Humanities, and Sociology, Vol. 8, No. 4, pp. 231-250, Apr, 2018.
DOI: https://dx.doi.org/10.21742/AJMAHS.2018.04.01

[4] Vladimir Vujović, Mirjana Maksimović, "Raspberry Pi as a Sensor Web node for home automation," Computers & Electrical Engineering, Vol. 44, pp. 153-171, May 2015.
DOI: https://doi.org/10.1016/j.compeleceng.2015.01.019

[5] Jeong-Joon Kim, Kwang-Jin Kwak, Don-Hee Lee, Yong-Soo Lee, "Study of Trust Bigdata Platform," Journal of The Institute of Internet, Broadcasting and Communication, Vol. 16, No. 6, pp. 225-230, Dec, 2016.
DOI: https://doi.org/10.7236/JIIBC.2016.16.6.225

[6] Zhendong Cheng, Zhongzhi Luan, You Meng, Yijing Xu, Depei Qian, Alain Roy, Ning Zhang, Gang Guan, "ERMS: An Elastic Replication Management System for HDFS," IEEE International Conference on Cluster Computing Workshops, pp. 32-40, Sept, 2012
DOI: https://doi.org/10.1109/ClusterW.2012.25

[7] Jens Dittrich, Jorge-Arnulfo Quiané-Ruiz, "Efficient big data processing in Hadoop MapReduce," Journal of the VLDB Endowment, Vol. 5, No. 12, pp. 2014-2015, 2012.
DOI: https://doi.org/10.14778/2367502.2367562

[8] Dong-Jin Shin, Jong-Min Eun, Ho-Geun Lee, Myoung Gyun Lee, Jeong-Min Park, Jeong-Joon Kim, "Big Data-based Log Collection and Analysis in IoT Environments," Journal of Engineering and Applied Sciences, Vol. 13, No. 5, pp. 1064-1072, May 2018.
DOI: http://dx.doi.org/10.3923/jeasci.2018.1064.1072

저자 소개

신 동 진(준회원)

• Dong-Jin Shin received his BS in Engineering at Korea Polytechnic University in 2018. He is currently a Master's course in the department of Smart Manufacturing Engineering at Korea Polytechnic University. His research interests include Big Data, Internet of Things(IoT), Network Security, etc.

박 지 훈(준회원)

• Ji-Hun Park is currently studying at Korea Polytech University for a bachelor's degree in computer engineering in 2018. His research interests include the Database systems, Big Data, Data Analysis, etc.

김 주 호(준회원)

• Ju-Ho Kim is currently studying at Korea Polytech University for a and is attending computer engineering. His research interests include Big Data, Data Processing, etc.

곽 광 진(준회원)

• Kwang-Jin Kwak received his MS in computer Science at Konkuk University in 2010 and 2016, He is currently a studying at Korea Polytech University for a doctor's course in Department of Smart Manufacturing Engineering. His intersts GIS, Information Retrieval, Text Mining, Database, NoSQL, etc.

박 정 민(정회원)

• Jeong-Min Park received his BS in Computer Science at Korea Polytechnic University in 2003. He received his MS and PhD in at SungKyunKwan University in 2005 and 2009, respectively. He is currently a professor at the department of Computer Science at Korea Polytechnic University. His research interests include Cyber Physical System(CPS), Autonomic Computing, Software Engineering, etc.

김 정 준(정회원)

• Jeong-Joon Kim received his BS and MS in Computer Science at Konkuk University in 2003 and 2005, respectively. In 2010, he received his PhD in at Konkuk University. He is currently a professor at the department of Computer Science at Korea Polytechnic University. His research interests include Database Systems, BigData, Semantic Web, Geographic Information Systems (GIS) and Ubiquitous Sensor Network (USN), etc.

※이 성과는 2018년 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2017R1A2B4011243).