**빅데이터 처리 및 응용**

**과제 보고서01**

- Hadoop 또는 Spark를 사용해 심야 시간

서울시 버스 정류장 분석-

날짜 : 2019.10.07

팀원 :

**<<시스템 구성도>>**

Mapper 에서 Key를 라인 번호, value를 라인으로 input을 받아서 라인을 스페이스 바 단위로 파싱하였다. 파싱 후, 네 번째 조각을 통화위치(정류장), 첫 번째 조각을 사용자ID로 각각 output의 키, 값으로 설정하고 출력하였다.

Reducer에서 앞선 Mapper의 output을 각각 input의 키, 값으로 받고 값으로 들어오는 사용자ID를 공백을 사이사이 껴 넣으며 나열하여 String형 변수에 이어 붙였다. 통화위치(정류장)과 사용자ID를 이어 붙인 String형 변수를 각각 output의 키, 값으로 설정하고 출력하였다.

"1" : "6 01063725815 10:05 D" "D : 6"

"2" : "6 01063725815 01:12 C" "C : 6"

"3" : "6 01063725815 02:12 A" "A : 6"

"4" : "10 01098769419 05:49 D" "D : 10"

[K2, V2]

Map

[K1, V1]

"5" : "10 01098769419 03:44 A" "A : 10"

"6" : "1 01090302357 01:16 G" "G : 1"

"7" : "1 01090302357 02:37 A" "A : 1"

"8" : "10 01098769419 02:30 D" "D : 10"

"A" : <"6", "10", "1">

Shuffle/Sort

"C" : <"6"> "A" : "6", "10"

Reduce

"D" : <"6", "10", "10" > "D" : "6", "10"

"G" : <"1">

<<환경 설정 내용>>

**<core-site.xml>**

: 네임노드 위치 지정에 사용되는 파일.

<?xml version="1.0"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

<name>fs.default.name</name>

<value>hdfs://localhost:9000</value> //네임노드를 같은 서버의 9000번 포트에 실행시킴

</property>

</configuration>

**<hdfs-site.xml>**

: HDFS와 관련된 내용을 편집하는데 사용하는 파일.

<?xml version="1.0"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

<name>dfs.replication</name>

<vlaue>1</vlaue> //블록의 복제 수를 1로 지정.(데이터 노드가 하나이기 때문)

</property>

</configuration>

**<mapred-site.xml>**

: MapReduce 프레임워크와 관련된 내영을 편집하는데 사용되는 파일.

<?xml version="1.0"?>

<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>

<!-- Put site-specific property overrides in this file. -->

<configuration>

<property>

<name>mapred.job.tracker</name> //잡 트래커의 위치는 같은 서버의 포트번호 9001에 지정

<value>localhost:9001</value>

</property>

</configuration>

<<프로그램 소스>>

<<소스 코드>>

<homewokr1Mapper.java>

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.LongWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Mapper;

public class homework1Mapper extends

Mapper<LongWritable, Text, Text, Text> {

//input 키: 라인 번호, 값: 라인

//output 키: 통화위치(정류장), 값: 사용자ID

private Text station = new Text(); //output 키

private Text userID = new Text(); //output 값

public void map(LongWritable key, Text value, Context context)

throws IOException, InterruptedException {

String st = value.toString(); //Text형을 String형으로 변환

String[] splits = st.split("\\s"); //스페이스바를 기준으로 파싱

station.set(splits[3]); //통화위치(정류장)을 output 키로 설정

userID.set(splits[0]); //사용자ID를 output 값으로 설정

context.write(station, userID); //output 키, 값 출력

}

}

<homework1Reducer.java>

import java.io.IOException;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Reducer;

public class homework1Reducer extends

Reducer<Text, Text, Text, Text> {

//input 키: 통화위치(정류장), 값: 사용자ID

//output 키: 통화위치(정류장), 값: 사용자ID 나열한 문장

private Text result = new Text(); //output 값

public void reduce(Text key, Iterable<Text> values, Context context)

throws IOException, InterruptedException {

String value = "";

for (Text val : values) { //사용자ID를 나열하기 위한 반복문

value += val.toString(); //Text형을 String형으로 변환하고 미리 선언해둔 value 변수에 추가

value += ' '; //value 변수에 공백 추가하여 사용자ID가 각각 분리되어 보이게 처리

}

result.set(value); //사용자ID 나열한 문장을 output 값으로 설정

context.write(key, result); //output 키, 값 출력

}

}

<homework1.java>

import org.apache.hadoop.conf.Configuration;

import org.apache.hadoop.fs.Path;

import org.apache.hadoop.io.IntWritable;

import org.apache.hadoop.io.Text;

import org.apache.hadoop.mapreduce.Job;

import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;

import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;

import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class homework1 {

public static void main(String[] args) throws Exception {

Configuration conf = new Configuration();

if (args.length != 2) {

System.err.println("Usage: homework1 <input> <output>");

System.exit(2);

}

Job job = new Job(conf, "homework1");

job.setJarByClass(homework1.class);

job.setMapperClass(homework1Mapper.class); //맵 클래스 설정

job.setReducerClass(homework1Reducer.class); //리듀서 클래스 설정

job.setInputFormatClass(TextInputFormat.class); //input 포맷 설정

job.setOutputFormatClass(TextOutputFormat.class); //output 포맷 설정

job.setOutputKeyClass(Text.class);

job.setOutputValueClass(Text.class);

FileInputFormat.addInputPath(job, new Path(args[0]));

FileOutputFormat.setOutputPath(job, new Path(args[1]));

job.waitForCompletion(true);

} }