J. Inf. Technol. Appl. Manag. 24(4): 187~196, December 2017
https://doi.org/10.21219/jitam.2017.24.4.187

ISSN 1598-6284 (Print)
ISSN 2508-1209 (Online)

하둡과 순차패턴 마이닝 기술을 통한 교통카드 빅데이터 분석

김우생* · 김용훈** · 박희성*** · 박진규****

Analysis of Traffic Card Big Data by Hadoop and Sequential Mining Technique

Woosaeng Kim* · Yong Hoon Kim** · Hee-Sung Park*** · Jin-Kyu Park****

Abstract

It is urgent to prepare countermeasures for traffic congestion problems of Korea's metropolitan area where central functions such as economic, social, cultural, and education are excessively concentrated. Most users of public transportation in metropolitan areas including Seoul use the traffic cards. If various information is extracted from traffic big data produced by the traffic cards, they can provide basic data for transport policies, land usages, or facility plans. Therefore, in this study, we extract valuable information such as the subway passengers' frequent travel patterns from the big traffic data provided by the Seoul Metropolitan Government Big Data Campus. For this, we use a Hadoop (High-Availability Distributed Object-Oriented Platform) to preprocess the big data and store it into a Mongo database in order to analyze it by a sequential pattern data mining technique. Since we analysis the actual big data, that is, the traffic cards' data provided by the Seoul Metropolitan Government Big Data Campus, the analyzed results can be used as an important referenced data when the Seoul government makes a plan about the metropolitan traffic policies.

Keywords : Big data analysis, Hadoop, Mongo database, Data mining

Received : 2017. 10. 16. Revised : 2017. 12. 27. Final Acceptance : 2017. 12. 27.

※ The present Research has been conducted by the Research Grant of Kwangwoon University in 2017 and the 13th KWIX(Kwangwoon ICT Exhibition).

* Corresponding Author, Professor, Computer Software Department, Kwangwoon University, 20 Kwangoon-ro, Nowon-gu, Seoul, 01897, Korea, Tel : +82-2-940-5217, e-mail : kwsrain@gmail.com
** Computer Software Department of Kwangwoon University, e-mail : hia5314@gmail.com
*** Computer Software Department of Kwangwoon University, e-mail : gmltjd0911@gmail.com
**** Computer Software Department of Kwangwoon University, e-mail : jinkyu2005@naver.com

1. 서 론

많은 인구와 다양한 경제・사회・문화 활동 및 시설이 응집되어 있는 세계 모든 도시들은 늘어나는 교통량으로 심각한 교통 혼잡 문제를 겪고 있으며, 교통 혼잡으로 야기되는 막대한 경제・사회・환경적 손실과 시민 건강 문제를 야기하고 있다. 특히 우리나라의 서울을 포함한 수도권 지역은 전체 인구의 46% 거주하고 있으며, 경제・사회・문화・교육 등 중심 기능들이 과도하게 집중되어 있어 세계 어느 도시보다도 심각한 교통 혼잡 현상을 보이며, 이로 인해 유발되는 다양한 교통 문제의 개선을 위한 대책 마련이 시급하다고 본다. 이러한 교통 문제 해결을 위한 정책 입안을 위해서는 도시의 교통 흐름을 이해하고, 도시 내 시민들의 통행 패턴과 통행 행태에 대한 정확한 파악이 요구된다[Heo, 1993].

서울을 중심으로 하는 수도권에서는 대중교통 이용자의 대부분이 교통카드를 이용하고 있으며, 이용자들로부터 생성되는 교통카드 빅데이터는 개개인의 통행에 대한 출발 지점과 최종 목적지, 이용 교통수단, 환승에 대한 위치와 시간이 기록되어 있다. 이러한 교통카드 빅데이터를 효과적으로 처리하면 수도권 지역에서의 통행의 패턴 등 다양한 정보를 추출해 낼 수 있으며, 이는 교통 정책이나 토지 이용 계획 및 시설 계획에 귀중한 기초 자료를 제공할 수 있다. 최근 정보 기술의 발전으로 교통카드와 같은 방대한 양의 데이터가 생산됨에 따라, 이러한 대용량의 데이터로부터 유용한 정보들을 추출하는 빅데이터 분석에 대한 요구가 증대되고 있다. 최근에 빅데이터를 처리하기 위한 많은 기술이 개발되는데, 대표적인 기술이 하둡(Hadoop : High Availability Distributed OO Platform)[1]과 NoSQL(Not Only SQL)[2]이다. 하둡은 빅데이터 처리를 위한 프레임워크로 분산 병렬 처리 시스템인 맵리듀스와 분산 파일 시스템인 HDFS(Hadoop File System)로 구성된다. 맵리듀스 프로그램 모델은 맵과 리듀스라는 두 개의 메서드로 구성된다. 맵 메서드는 키와 값으로 구성된 데이터를 입력 받아 이를 가공한 후 새로운 키와 값으로 구성된 목록을 출력하고, 리듀스 메서드는 새로운 키로 그룹핑 된 값의 목록을 입력 받아 값의 목록에 대한 집계 연산을 실행해 새로운 키와 값으로 구성된 목록을 생성한다. 반면 NoSQL은 전통적인 관계형 데이터베이스 보다 덜 제한적인 일관성 모델을 이용한 분산 환경에서 주로 빅데이터의 저장 및 관리에 사용된다. NoSQL의 한 제품인 몽고 데이터베이스는 강력하고 유연하며 확장성이 높은 문서 지향의 데이터베이스로, 내장 문서와 배열 따위의 표현이 가능해서 복잡한 객체의 계층 관계를 하나의 레코드로 표현할 수 있다.

본 연구에서는 교통카드 빅데이터로부터 지하철 승객들이 자주 사용하는 정류장의 시퀀스와 시간대별의 승객 수 등의 정보를 분석하고자 한다. 이를 위하여 하둡으로 빅데이터에 대한 전처리를 수행하여 몽고 데이터베이스에 지하철 승객들의 정류장 시퀀스를 저장한 후 순차패턴 데이터 마이닝 기법을 적용해 승객들의 빈발 통행 패턴 등을 찾아낸다. 본 연구가 기존의 연구들과 다른 점은 하둡 등의 빅데이터 처리 기술을 이용해 지하철 승객들의 빈발 통행 패턴을 처음으로 구하였고, 또한 실제적인 빅데이터인 서울시 빅데이터 캠퍼스에서 제공하는 교통카드 환승 데이터를 분석하였기에, 이러한 분석 기법과 결과는 앞으로 대중교통 체계의 발전에 활용 가능하다고 본다.

본 논문은 다음과 같이 구성된다. 제 2장은 교통

1) http://hadoop.apache.org.

2) http://nosql-database.org.

카드 분석과 관련된 연구에 대해서 소개하며 제 3장은 하둡을 통한 교통카드 빅데이터 전처리와 결과를 몽고 데이터베이스에 저장하는 방법, 그리고 순차패턴 마이닝 기법을 통해 승객들의 빈발 통행 패턴을 분석하는 방법을 설명한다. 제 4장은 서울시 빅데이터 캠퍼스에서 제공하는 교통카드 환승 데이터에 하둡과 순차패턴 마이닝 기법을 적용하여 승객들의 빈발 통행 패턴과 승객 수 등의 분석 결과를 보이며, 마지막으로 제 5장에서 결론과 향후 연구 과제에 대해서 언급한다.

2. 관련 연구

하둡 등의 빅데이터 기술을 사용하지 않은 기존의 교통 관련 빅데이터 분석에 대한 연구로는, 대용량 교통카드를 이용하여 대중교통 이용자들이 만들어 내는 통행 거래 자료를 바탕으로 통행 행태와 통행 흐름의 공간적 특징을 분석하는 연구[Lee and Park, 2005], 대용량의 교통카드 트랜잭션 데이터베이스에서 순회 패턴 탐사에 기반한 데이터 마이닝 기법을 사용하여 통행 패턴의 공간적 특징과 시점 간 차이를 분석한 연구[Park and Yoon, 2001; Lee and Park, 2006], 승객들의 통행 패턴에 대한 분석 연구를 확장해 일 년에 하루 씩 2004년에서 2006년까지 3일간의 교통카드 트랜잭션 데이터베이스로부터 승객 시퀀스의 평균 정류장 개수와 환승 횟수 등을 연도별로 비교한 연구가 있다[Park and Lee, 2007]. 또한 교통카드 트랜잭션 데이터베이스에서 지하철 승객들의 하루 동안의 통근 패턴들 중에서 많은 승객들이 이용할 것이라고 예측되는 8가지 패턴들을 탐사하고, 이 패턴들의 각 경우에 대한 승객들의 수와 승하차 시간을 교통카드 트랜잭션 데이터베이스에서 추출하는 연구[Park, 2010], 서울 지하철의 주요 링크상의 승객 흐름을 분석하기 위해, 2007년도 하루 동안에 승객들이 사용한 약 1,100만 건의 교통카드 트랜잭션들로 이루어진 데이터베이스로부터 지하철역과 인접한 지하철역 사이의 각 링크 승객 흐름을 시간대별로 보여주는 연구가 있다[Park and Lee, 2010].

반면에 하둡 등의 빅데이터 처리 기법을 적용한 교통 관련 빅데이터 분석에 대한 연구로는, 교통 데이터에 맵리듀스를 적용해 요약한 교통 패턴을 NoSQL 데이터베이스에 저장한 후, 질의로 교통 체증에 대한 추세를 추출하는 연구가 있으며[Titus et al., 2014], 교통량 데이터를 맵리듀스를 사용해 행렬로 변환한 후 몽고 데이터베이스에 저장하고, 효율적인 교통량 질의를 수행하기 위해 행렬로부터 계산된 SVD(singular value decomposition)를 이용하는 연구가 있다[Titus et al., 2014; Titus et al., 2015]. 또한 대량의 연관된 데이터들로 구성된 도로 수송 관리 정보를 분석하기 위해 연관 규칙 데이터 마이닝 기술을 맵리듀스 환경에서 수행할 수 있는 방법에 관한 연구가 있으며[Zhu et al., 2013], 교통사고의 발생빈도를 줄이기 위해서는 교통사고의 발생원인 및 운전자 주의뿐만이 아니라 사고 당시의 환경과 도로 상황 등 특징을 분석하고 규명하는 작업이 필요하기 때문에, NoSQL 기반 Map-Reduce 프로그래밍 모델을 이용하여 교통 사건의 빅 데이터를 분석하여 위치, 시기, 규모, 날씨, 도로 현황, 운전자 특성 등의 관점에서 효율적으로 예방할 수 있는 방법에 대한 연구가 있다[Zheng and Wang, 2014].

3. 교통카드 빅데이터 전처리, 관리 및 분석

3.1 교통카드 빅데이터

수도권 지역의 승객이 버스, 지하철 등 대중교통을 이용하면서 교통카드를 사용하는 경우에

(주)한국 스마트카드(KSCC)에서 요금을 정산한다. KSCC에서 관리하는 승객들의 트랜잭션은 승객의 승차와 하차에 관련된 여러 가지 속성들을 가지고 있다. 본 논문에서는 서울시 지하철 승객들의 빈발 통행 패턴 등을 구하기 때문에 승객들의 트랜잭션으로부터 승차역, 하차역 등 일부 속성들만을 이용한다. <Table 1>은 본 논문과 관련된 교통카드번호, 트랜잭션ID, 환승 횟수, 승차일시, 승차역, 하차일시, 하차역의 속성들과 일부 트랜잭션들을 보여 준다. 예를 들어, 트랜잭션 1은 어떤 지하철 승객이 2007년 5월 16일 오전 5시 18분에 1호선 청량리역(239)에서 승차하여 오전 5시 41분에 시청역(202)에서 하차하는 내용이다. 반면, 트랜잭션 2는 환승 승객인 경우로 2007년 5월 16일 오전 10에 1호선 신이문역(2516)에서 승차하여 오전 11시 20분에 서대문역(202)에서 하차하는 내용이다.

〈Table 1〉 Example of Traffic Card Transactions

TR\Attr	Traffic Card Num, TR ID, Transfer Num, Boarding Time, Entraining St, Departing St
TR 1	658, 015, 0, 200705160518, 239, 200705160541, 202
TR 2	656, 012, 1, 200705161000, 2516, 200705161120, 1456

3.2 교통카드 빅데이터 전처리 및 데이터베이스를 통한 관리

본 논문에서는 <Table 1>과 같은 트랜잭션이 하루에도 천만 건 이상을 포함하는 대용량 트랜잭션들에서 빈발 통행 패턴 등을 추출하고자 한다. 따라서 본 연구에서는 하둡의 맵리듀스 기술을 적용해 교통카드 빅데이터로부터 승객들의 통행 패턴 즉, 정류장 시퀀스를 구한다. 하둡은 구조화 되지 않은 데이터 처리를 위해 키와 값으로 이루어진 쌍을 데이터의 기본 구조로 사용하며 다음의 단계를 수행한다. 하둡은 하둡 파일 시스템(HDFS)의 입력 데이터 파일을 여러 개로 분할(split)하고, 각 분할 데이터를 키/값들로 변환한 후, 각 키/값(key1, val1)을 입력으로 맵 함수를 실행한다. 맵 함수는 입력값을 변형한 후 키/값 목록(List(key2, val2))을 서버의 로컬 저장소에 저장한다. 다음으로 하둡은 키를 중심으로 정렬해 같은 키를 갖는 값들을 그룹핑 한 후, 각 키와 값의 목록(key2, List(val2))을 입력으로 리듀스 함수를 실행한다. 리듀스 함수는 입력값을 집계한 후 키/값 목록(list(key3, val3))을 최종 저장소로 출력한다.

하둡을 통하여 교통카드 빅데이터로부터 승객들의 정류장 시퀀스 등을 추출하는 프로그램은 매퍼 클래스, 리듀서 클래스와 메인 메서드로 이루어진다. <Figure 1>은 매퍼 클래스의 수도 코드로 교통카드 빅데이터로부터 승차일시와 승차역과 하차역 코드 번호를 추출한다. 여기서, 입력은 레코드 번호(key1)와 레코드(val1)이며, 출력은 승차일시(key2)와 승차역과 하차역 코드 번호(val2)의 목록이다.

```
Class TraffMapper extends Mapper
        <LongWritable, Text, Text, Text>
1. Create and initialize variable key2 and val2 : Map
   (key1, val1)
2. ExtractFrom_Record(Boarding Time, Entraining
   St,Departing St)
3. key2 := Boarding Time
4. val2 := Entraining St & Departing St
5. emit(key2, val2)
```

〈Figure 1〉 Pseudo-code of Mapper Class

반면 <Figure 2>는 리듀서 클래스의 수도 코드로 "정류장 시퀀스" 컬렉션(collection)에 삽입할 각 문서를 출력한다. 여기서, findRoute() 함수는 승차역과 하차역을 이용해 중간 정류장들을 찾는 함수이며, 입력은 승차일시(key2)와 승차역과 하차역 코드 번호(val2)의 목록이며, 출력은 문서 id(key3)와 문서 내용(val3)의 목록이다.

```
Class TraffReducer extend Reducer
<Text, Text, BSONWritable, BSONWritable>
1. Create and initialize variable PassengerNum and StationSeq
2. Function findRoute() Reduce(key2, List(val2))
3. ExtractFrom_ key2
   (Boarding Year, Boarding Month, Boarding Day,
   Boarding Time)
4. id := {year : Boarding Year, month : Boarding Month,
        day : Boarding Day, hour : Boarding Time)}
5. For each Entraining St & Departing St in List(val2) do
6.     List<StationSeq>.add(findRoute (Entraining St &
       Departing St))
7.     PassengerNum++;
8. DocContent :={PassengerNum, List<StationSeq>}
9. emit(id, DocContent)
```

〈Figure 2〉 Pseudo-code of Reducer Class

```
{//each document has an unique ID
id: {year : Boarding Year, month : Boarding Month,
     day : Boarding Day, hour : Boarding Time},
Passenger Num : n,
Station Sequence List : [ // [ ] represents array
{St 1, St 2, … , St m1},
{St 1, St 2, … , St m2},
          …,]
};
```

〈Figure 3〉 One Document of "Station Sequence" Collection

하둡의 실행 결과는 일반적으로 하둡 파일시스템(HDFS)에 만들어진다. 그러나 본 논문에서는 하둡의 결과를 즉, 리듀스 함수의 출력을 직접 몽고 데이터베이스에 저장해 관리하도록 한다. 즉, 하둡의 결과를 몽고 데이터베이스로 보내기 위해서는 몽고 데이터베이스를 하둡의 입력 소스 또는 출력 목적지로 연결해 주는 몽고 데이터베이스 커넥터 설치가 필요하다.[3] 이를 위하여 하둡의 main 메서드에서는 Configuration 클래스를 이용하여 mongo.out.uri을 설정한 뒤 해당 Configuration 클래스를 이용해 Job 객체를 만들고, 출력 파일의 포맷은 MongoOutputFormat.class로 그리고 리듀스의 출력 키/값 타입은 BSONWritable.class로 설정한다.

3) https://github.com/mongodb/mongo-hadoop.

또한 몽고 데이터베이스의 CRUD (create, read, update, delete) 연산을 사용하여 "정류장 시퀀스" 컬렉션을 생성한다. <Figure 3>은 "정류장 시퀀스" 컬렉션의 한 문서이며, 특정 시간대의 승객 n명의 정류장 시퀀스 리스트는 문서에 내포되는 것을 알 수 있다.

3.3 교통카드 빅데이터 분석

<Table 2>는 "정류장 시퀀스" 컬렉션에 포함된 한 문서의 예로써, 2016년 1월 10일 오전 9시에서 오전 10시 사이의 전체 승객은 5명이고, 각 승객은 서로 다른 정류장 시퀀스를 만들어 내고 있다. 예를 들어, 첫 번째 승객은 1번 정류장에서 승차하여, 2번, 3번, 5번 정류장을 거쳐서 4번 정류장에서 하차하는 것을 나타내고 있다.

〈Table 2〉 Example of Station Sequence Database

Boarding Time	Passenger Num	Station Sequence List
2016010109	5	1 2 3 5 4 1 2 3 5 4 1 2 3 3 2 1 5 4 2 3 2 3 5 4 1 2 5 4 1 2 3 5 4 2 3

한 시퀀스의 지지도는 데이터베이스에서 그 시퀀스를 포함하는 트랜잭션들의 개수이다. 사용자가 정해 준 지지도보다 많은 시퀀스를 포함하는 패턴을 빈발 시퀀스라고 부르며, 길이가 K인 빈발 시퀀스의 집합을 F_K로 표기한다. 반면 빈발 시퀀스가 될 가능성이 있는 패턴을 후보 시퀀스라고 부르며 길이가 K인 후보 시퀀스의 집합을 C_K로 표기한다. 또한 한 시퀀스가 다른 시퀀스의 부분 시퀀스가 아닐 경우 최대 시퀀스라고 한다. 기존의 순차패턴 탐사 알고리즘인 GSP (Generalized Sequential Pattern)를 통해 빈발 시퀀스를 찾는 방법은 먼저 후보 시퀀스를 찾아낸 뒤, 이들 중 사용자가 정해 준 최소 지지도 이상인

시퀀스만을 모아 빈발 시퀀스를 찾아 나간다.[4] 따라서 크기가 1인 후보 집합부터 시작해 빈발 집합을 찾아 가다가($C_1 \rightarrow F_1 \rightarrow C_2 \rightarrow F_2 \rightarrow C_3 \cdots$), 어느 순간 F_k가 하나도 발견되지 않거나 F_k로부터 C_{K+1}을 더 이상 만들어 낼 수 없을 때 GSP 알고리즘은 종료한다.

기존의 순차패턴 탐사에서는, 예를 들어, (4 → 5)나 (4 → 6)은 (4 → 5 → 6)의 부분 시퀀스가 되나, 통행 패턴 탐사의 경우는 항목의 순서 유지뿐만 아니라 항목들이 연속적으로 발생해야 하기 때문에 (4 → 6)은 (4 → 5 → 6)의 부분 시퀀스가 아니다. 따라서 통행 패턴 탐사 문제는 순차패턴 탐사의 특별한 경우로 볼 수 있다. 따라서 본 논문에서는 기존의 GSP에 이러한 점을 반영하여 다음과 같은 방법으로 빈발 통행 패턴을 찾는다. 예를 들어, <Table 2>의 정류장 시퀀스 데이터베이스와 80%의 최소 지지도가 주어졌을 때, 빈발 시퀀스를 찾아내는 과정이 <Table 3>에서 <Table 6>까지 나와 있다. 빈발 1-시퀀스(F_1)를 찾기 위한 후보 1-시퀀스(C_1)는 <Table 2>의 모든 항목들이 된다. 이들 중 <Table 2>에서 최소 지지도를 만족하는 F_1은 <Table 3>와 같다. 빈발 2-시퀀스(F_2)를 찾기 위한 후보 2-시퀀스(C_2)는 <Table 3>의 $F_1 \times F_1$에서 자기 자신과 같은 2-시퀀스를 제외한 20개의 시퀀스들이다. 이들 중 <Table 2>에서 최소 지지도를 만족하며 항목들이 연속적으로 발생하는 F_2들은 <Table 4>와 같다. 빈발 3-시퀀스(F_3)를 찾기 위한 후보 3-시퀀스(C_3)를 찾는 방법은 <Table 4>에서 하나의 F_2 두 번째 항목과 다른 하나의 F_2 첫 번째 항목이 동일한 경우에(예를 들어, 2 → 3 & 3 → 5), 두개의 F_2를 결합하여 한 개(2 → 5)의 C_3을 만든다. 이들 중 <Table 2>에서 최소 지지도를 만족하며 항목들이 연속적으로 발생하는 F_3은 <Table 5>과 같다. 비슷한 방법으로 최소 지지도를 만족하며 항목들이 연속적으로 발생하는 빈발 4-시퀀스(F_4)는 <Table 6>과 같다. 후보 5-시퀀스(F_5)는 <Table 6>로부터 만들 수가 없기 때문에 알고리즘은 여기서 중지한다. 모든 빈발 시퀀스들 중에서 서로 간에 중복되는 것(예를 들어, F_3의 2 → 3 → 5는 F_4의 2 → 3 → 5 → 4에 포함됨)을 제외한 즉, 어떤 다른 시퀀스에도 포함되지 않는 최대 빈발 시퀀스는 <Table 7>과 같다. 즉, 2016년 1월 10일 오전 9시에서 10시 사이의 통행 패턴 중 80%의 지지도를 만족하는 빈발 통행 패턴은 1 → 2와 2 → 3 → 5 → 4이다.

〈Table 3〉 F_1 Sequence

Frequent 1-Sequence F_1	
1	5
2	5
3	5
4	5
5	5

〈Table 4〉 F_2 Sequence

Frequent 2-Sequence F_2	
1 → 2	4
2 → 3	5
3 → 5	4
5 → 4	5

〈Table 5〉 F_3 Sequence

Frequent 3-Sequence F_3	
2 → 3 → 5	4
3 → 5 → 4	4

〈Table 6〉 F_4 Sequence

Frequent 4-Sequence F_4	
2 → 3 → 5 → 4	4

〈Table 7〉 Maximum Frequent Sequence

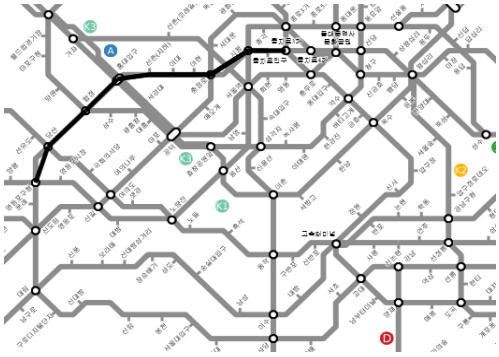
Maximum Frequent Sequence	Support
1 → 2	4
2 → 3 → 5 → 4	4

4) https://en.wikipedia.org/wiki/GSP_algorithm.

4. 교통카드 빅데이터 분석 결과

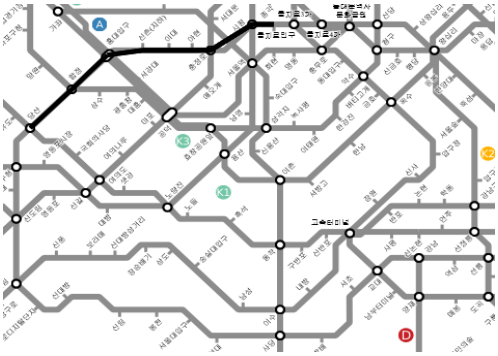
서울시 빅데이터 캠퍼스에서 제공하는 서버는 우분투 16.04가 설치된 I5-6600K, RAM 16GB, SSD 512GB의 PC로 VM 5대의 가상 분산 모드로 사용하였다. 사용한 하둡 버전은 2.6.4로 하둡 디렉토리 내부의 Hadoop-env.sh, core-site.xml, hdfs-site.xml, mapred-site.xml, yarn-site.xml, masters, slaves 등의 파일들로 환경 설정을 하였고, SSH Key를 생성하여 하둡 간 통신을 수행하였다.

교통카드 빅데이터를 분석하기 위해 교통카드 환승 데이터 중 2017년 4월 28일의 데이터를 사용하였다. <Figure 4>는 17.04.28일 전체 시간대에 대한 5%의 지지도를 만족하는 빈발 통행 패턴들을 나타낸다. 총 9개의 빈발 통행 패턴은 을지로 3가에서 영등포구청 사이의 노선들이었으며, 이 노선들을 모두 연결하여 하나의 굵은 실선으로 표시하였다.



〈Figure 4〉 5% Support of Frequent Travel Pattern During the whole Time of 17. 04. 28

<Figure 5>는 17.04.28일 18시의 데이터의 지지도 10%를 만족하는 빈발 통행 패턴을 나타내며 <Figure 4>의 결과와 매우 유사함을 알 수 있다. 당일에 지하철을 이용한 승객 수를 그래프로 표시한 <Figure 7>에 의하면 출퇴근 시간대의 이용 승객 숫자가 가장 많은 것을 알 수 있다. 따라서 을지로 입구에서 당산 사이의 노선 부근에 이용 승객들의 직장이나 저녁에 들리는 장소가 많이 있을 것으로 추정된다.

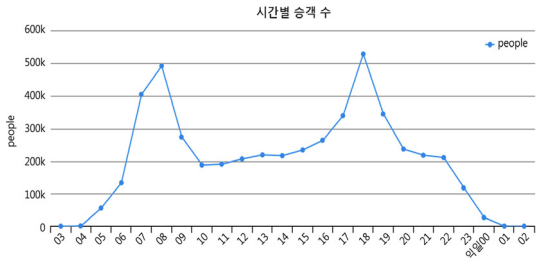


〈Figure 5〉 10% Support of Frequent Travel Pattern During the 6 p.m. of 17. 04. 28

<Figure 6>은 17.04.28 일 14시의 데이터의 지지도 3%를 만족하는 빈발 통행 패턴을 보이나, <Figure 4>나 <Figure 5>와는 겹치는 빈발 통행 패턴이 나타나지 않았으며 해당 구간의 지지도도 낮은 것을 알 수 있다. 해당 결과들을 분석해 볼 때, 출퇴근하는 승객들의 이용 데이터가 출퇴근 시간대 뿐 아니라 전체 시간대에서의 빈발 통행 패턴 결과에 큰 영향을 끼침을 알 수 있으며, 다른 시간대에서의 승객 이용률은 많지 않고 불규칙한 패턴을 보임을 알 수 있다.

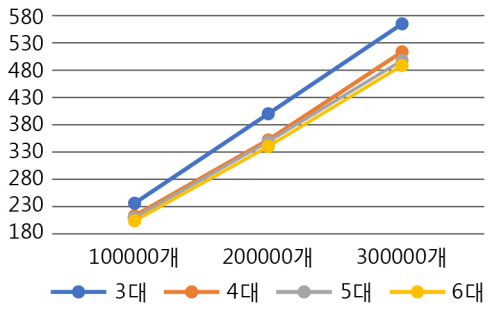
〈Figure 6〉 3% Support of Frequent Travel Pattern During the 2 p.m. of 17. 04. 28

<Figure 7>은 17. 04. 28일 03시부터 17. 04. 29일 02시까지의 승객 수를 그래프로 표시한 차트이다. 평일이기 때문에 출퇴근하는 사람들이 많으므로 출근 시간대인 7~8시와 퇴근 시간대인 17~19시의 전철 이용승객 숫자가 급격히 늘어나는 것을 확인할 수 있다. 이 결과는 앞의 <Figure 4>, <Figure 5>, <Figure 6>의 결과를 설명해 준다고 볼 수 있으며, 출근 시간인 7~8시와 퇴근 시간인 17~19시의 경우의 이용 데이터는 전체 데이터의 약 43%를 차지하며 출퇴근 시간의 데이터가 전체 데이터에 영향을 얼마나 끼치는지 실제 수치로 확인 할 수 있다. 이 결과는 또한 기존의 연구[Park and Lee, 2010]와도 비슷한 결과로, 측정 날짜와 상관없이 출퇴근 시간대의 전철 이용 승객 수가 가장 많음을 알 수 있다.



<Figure 7> Number of Passenger from 3 a.m. of 17. 04. 28 to 2 a.m. of 17. 04. 29

하둡의 분산 환경에서의 성능을 비교하기 위해 서버의 대수를 달리하여 분산 환경에서 수행하였다. <Figure 8>의 X축은 테스트한 트랜잭션의 개수를 나타내며 Y축은 시간(초)를 나타낸다. 서버의 대수가 늘어날수록 같은 데이터를 더 적은 시간에 수행하는 것을 알 수 있다. 다만 서버의 대수가 4대 이상에서는 대수가 늘어나도 성능상의 많은 차이가 나지 않는 것은, 사용한 데이터가 아주 크지는 않기 때문에 많은 서버의 이점이 제대로 활용되지 못하기 때문인 것으로 사료된다. 하지만 그러한 환경에서도 데이터의 크기가 커질수록 성능의 차이가 벌어지는 것을 알 수 있다.



<Figure 8> Performance Comparison According to the Number of the Nodes

5. 결 론

우리나라의 서울을 포함한 수도권 지역은 경제・사회・문화・교육 등 중심 기능들이 과도하게 집중되어 있어 세계 어느 도시보다도 심각한 교통 혼잡 현상을 보이며, 이로 인해 유발되는 다양한 교통 문제의 개선을 위한 대책 마련이 시급하다. 서울에서는 2004년 대중교통체계 개편과 함께 정보 기술을 도입한 교통카드 사용이 활성화되었고, 특히 자료 안에는 개개 통행자의 통행에 대한 출발 지점과 목적지, 이용 교통수단, 환승에 대한 위치와 시간 등에 대한 정확한 정보가 담겨 있어 이를 적절히 분석하면 도시 내 통행 흐름 및 통행 행태에 대한 다양한 정보를 파악해 낼 수 있게 되었다.

본 연구에서는 빅데이터 기술인 하둡을 사용하여 교통 카드 빅데이터를 전처리하고 몽고 데이터베이스에 저장한 후 통행 패턴을 고려한 순차패턴 데이터 마이닝 기법을 적용하여 승객들의 빈발 통행 패턴 등을 구하였다. 또한 실제적인 빅데이터 즉, 서울시 빅데이터 캠퍼스에서 제공하는 교통카드 환승 데이터를 분석하였기에 이를 기반으로 수도권 교통 정책을 입안할 때 중요한 참고 자료로 사용할 수 있다. 분석을 통하여 출퇴근 시간대의 승객들이 가장 많았으며, 이 시간대의 승객들에 의해 당일 전체 시간대의 빈발

통행 패턴도 형성됨을 알 수 있었다. 반면 다른 시간대에서의 승객 이용률은 많지 않고 불규칙한 패턴을 보임을 알 수 있었다. 특히 영등포구청에서 을지로 3가 사이의 노선이 가장 붐비기 때문에, 해당 노선에 배차 간격을 더 자주 한다면 승객들의 지하철 이용에 도움이 될 것으로 사료된다. 본 연구에서는 2017년 4월 28일 하루치의 데이터만을 사용하였으나, 추후 더 많은 데이터에 대한 분석을 통해 지하철 신규노선이나 역의 확장 등, 승객들의 편의를 위한 다양한 방안을 분석해볼 수 있을 것으로 사료 된다.

References

[1] Damaiyanti, T. I., Imawan, A., and Kwon, J. H., "Extracting Trends of Traffic Congestion Using a NoSQL Database", in Proceedings of 2014 IEEE Fourth International Conference on Big Data and Cloud Computing, 2014, pp. 209-213.

[2] Damaiyanti, T. I., Imawan, A., and Kwon, J. H., "Querying road traffic data from a document store", in Proceedings of 2014 IEEE Fourth International Conference on Utility and Cloud Computing, 2014, pp. 485-486.

[3] Damaiyanti, T. I., Imawan, A., Kwon, J. H., and Choi, Y.-H., "Implementation of a road traffic data query system using NoSQL Database", DB SIGDB, 2015.

[4] Heo, W., "Commuting Flows of Seoul : Geographical characteristic and change", *Journal of Korean Society of Transportation*, Vol. 11, No. 1, 1993, pp. 5-21.

[5] Lee, K. and Park, J., "Travel Patterns of Transit Users in the Metropolitan Seoul", *Journal of the Economic Geographical Society of Korea*, Vol. 9, No. 3, 2006, pp. 379-395.

[6] Lee, K. and Park, J., "Traversal pattern analysis of transit users in the Metropolitan Seoul", Proceedings of International Forum on the Public Transportation Reform in Seoul, 2005.

[7] Park, J. and Lee, K., "Analysis of Passenger Flows in the Subway Transportation Network of the Metropolitan Seoul", *KIISE Transactions on Computing Practices*, Vol. 16, No. 3, 2010, pp. 316-323.

[8] Park, J. and Lee, K., "Mining Trip Patterns in the Large Trip-Transaction Database and Analysis of Travel Behavior", *Journal of the Economic Geographical Society of Korea*, 2007, pp. 44-63.

[9] Park, J. and Yoon, J., "A System for Mining Traversal Patterns from Web Log Files", Proceedings of the 28th KIISE Fall Conference, 2001.

[10] Park, J., "Mining Commuter Patterns from Large Smart Card Transaction Databases", Proceedings of KIISE Fall Conference, 2010.

[11] Zheng, X. and Wang, S., "Study on the method on road transport management information data mining based on pruning Eclat algorithm and mapreduce", Procedia-Social and Behavioral Sciences, 2014.

[12] Zhu, T., Yu, J., and Jiang, F., "Rtic-c : A big data system for massive traffic information mining", *in Proceedings of 2013 International Conference on Cloud Computing and Big Data*, 2013, pp. 395-402.

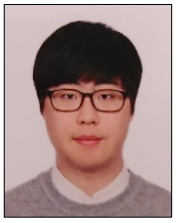
저자소개

김 우 생

Woosaeng Kim is currently a Professor of Computer Software department of Kwangwoon University. He received the bachelor's degree in the department of Computer Science from University of Texas at Austin. He received the MS and Ph.D. degree in the department of Computer Science from University of Minnesota. He had worked as a system engineer at Hyundai Electronics Co. His current research interests include database, multimedia, web application, etc.

김 용 훈

Yong Hoon Kim is currently a student of Computer Software department of Kwangwoon University.



박 희 성

Hee-Sung Park is currently a student of Computer Software department of Kwangwoon University.

박 진 규

Jin-Kyu Park is currently a student of Computer Software department of Kwangwoon University