Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 98 (2016) 554 – 559

Procedia Computer Science

The 3rd International Symposium on Emerging Information, Communication and Networks

Integration of Big Data for Connected Cars Applications Based on Tethered Connectivity

Lionel Nkenyereye, Jong-wook Jang *

*Dong-Eui University, 176 Eomgwangro,Busanjin-Gu, Busan,614-714,Korea

Abstract

The wireless communication technologies built-in or brought in the vehicle enable new in-car telematics services. The development of connected cars emphasizes the use of sophisticated computation framework for gathering, analyzing a large volume of data generated in all aspects of vehicle operations using Big Data technologies. Since these data are essential for many connected cars applications, the design and monitoring of MapReduce algorithms for processing vehicle's data using Hadoop framework will allow to build a hosting of analytics data source. This hosting data source allows different connected cars industry ecosystem to access useful data they need to afford connected cars applications.

This paper studies design steps to take in consideration when implementing MapReduce patterns to analyze vehicle's data in order to produce accurate useful data that are hosted at the automakers and connect cars services providers. Experiment results show that MapReduce join algorithm is highly scalable and optimized for distributed computing than Statistical Analysis System (SAS) framework and HiveQL declarative language.

Peer-review under responsibility of the Program Chairs

Keywords: Connected Car, remote on-line vehicle diagnostics system, MapReduce framework, Hadoop, Big data, Join algorithm

* Corresponding author. Tel.: +821056000845; fax: +82518901724.
E-mail address: jwjang@deu.ac.kr

Peer-review under responsibility of the Program Chairs
doi:10.1016/j.procs.2016.09.083

1. Introduction

The information technology authorizes information sharing between vehicle and driver, vehicle sensor operations and the vehicle surrounding, road condition[1]. Connected vehicles and advanced technology permit for operative, over internet Remote On-line Vehicle Diagnostics System (ROVDS)[2]. These information technology expand challenges of ROVDS that submerge currently acquisition of information from what is captured, processed and stored[3]. Such a system might store 40TB of status and conditions over a year in case connected vehicles reporting Diagnostic Troubles Codes (DTC) status in the size of 25 GB on a daily basis[3].

The collection of vehicle's data allows the third party interested in automobile ecosystem for continuously support new in-car telematics services. Automobile ecosystem is composed by car manufacturers, repair shops, road and transportations authorities .The new in-car telematics services increase safety driving for the driver but also provide for free or pay as the car owner subscribes to these telematics services. Therefore, in order to provide great market for affording these in-car telematics services, the reality of big data technology was explored at a recent Connected Vehicle Trade Association (CVTA)[3].

The main challenge for connected cars services providers is that the collection of same vehicle's data such as engine temperature, engine Revolutions per minute(RPM), vehicle speed are subjected to different connected cars applications which the final purpose of each of them differ as shown on the Fig. 1. The automakers and/or connected cars services providers have to design an efficient monitoring and analytics framework that allows MapReduce framework to be implemented with a highly analytics of vehicle' data in order to produce accurate useful information for decision making by the connected car industry ecosystem[4]. The Fig. 2. shows the overview of the hosting of analytics data using Big Data Technology[5].

The main contribution of this work states a novel analytical model based big data technology for monitoring vehicle's diagnostics data and building up a data warehouse accessible to automobile ecosystem. The framework analytical model is expressed as a function that takes input from the Hadoop Data File System (HDFS) on which the MapReduce join reduce side algorithm is applied to get the final output. The analysis keeps splitting out the vehicle's data based on events occurred inside the vehicle, then select the key value pairs to include in the input functions which are submitted to MapReduce jobs implemented on the Hadoop cluster.

2. Monitoring and analytics framework based on MapReduce

The monitoring and analytics framework is based on a data-driven approach. This approach consists of collecting data sets uploaded from connected cars. Those data are then monitored based on different aspects of activity of the vehicles that we quote as "Events". The first event relates the vehicle's movement and journey trip. The next step

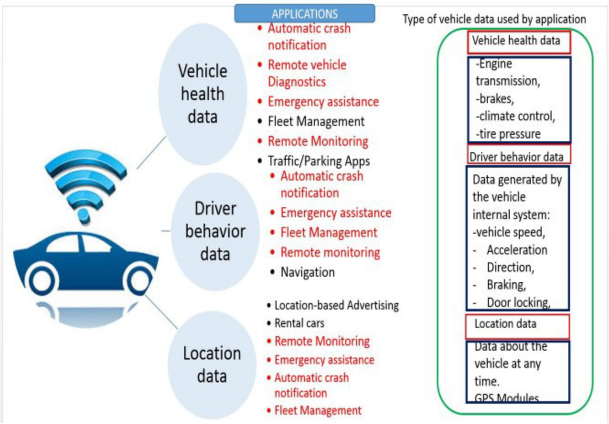


Fig.1.Vehicle's data that enable Connected Car applications

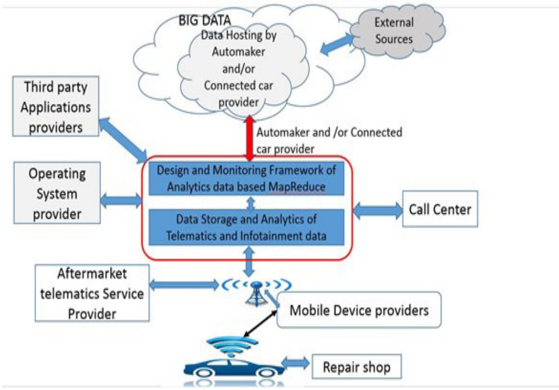

BIG DATA
Data Hosting by Automaker and/or Connected car provider
External Sources
Third party Applications providers
Automaker and /or Connected car provider
Operating System provider
Design and Monitoring Framework of Analytics data based MapReduce
Data Storage and Analytics of Telematics and Infotainment data
Call Center
Aftermarket telematics Service Provider
Mobile Device providers
Repair shop

Fig. 2. Overview of the hosting of analytics using Big data integration for Connected Car applications

related to that event is about collection of vehicle's diagnostic data while the driver is driving. The system design of the proposed analytics framework associates for both events vehicle's movement, journey trip, location based service, over speed, mileage and diagnostics of car's engine events an appropriate subset of information. The subset of information are journey data, Global Positioning System(GPS) data, driver behavior data based on the smartphone's in- built accelerometer, engine data and car diagnostics data. As shown on the Fig. 3. for instance, the analysis process takes the vehicle's movement and journey trip event, then associates in turn RPM value of the vehicle to detect if the engine is running, current data, and accelerometer data to detect vehicle's movement. For this event, the analysis process extracts the value of the RPM every three (3) minutes to detect the state of the vehicle, either is in idle state or not. These data are then uploaded to the Hadoop Data File System (HDFS). The data set on the HDFS serve as the basis for collecting the useful information to submit to MapReduce functions for processing.

The Hadoop framework splits the input data-set into multiple chunks, each of which is assigned a map task that can process the data in parallel. Each map task reads the input as a set of key-value pairs and produces a transformed set of key-value pairs as the output. The framework shuffles and sorts outputs of the map tasks, sending the intermediate key-value pairs to the reduce tasks, which group them into final results. For the vehicle's movement and journey trip the results are for example vehicle movement time, idling time, traveled time, journey trip time. These results are then stored on the hosting database with an additional field to indicate on which vehicle's telematics applications the output are intended to be applied .

The captured data sets are uploaded using tethered connectivity models that stands on the obligation of carrying the smart phone inside the vehicle. This smart phone is used as a modem via Wi-Fi. When on-board diagnostics data are uploading to the database, Apache Sqoop[6,7] performs a replication import of data required to run Map Reduce functions. Using HIVE[9] has an important role especially for data stored unto a relational database. Sqoop generates a Hive table based on table originally relational data source and at the same time stores data on HDFS[8,10]. One of the most key of Big Data technology in this paper is to take the collection of on-board diagnostics and process them according to the final output of useful information and in turn writes back the processing outcomes to the MySQL database. This is achieved by using HIVEQL and Map Reduce functions.

2.1 Distributed Computing with MapReduce

MapReduce is a framework for performing analysis of HDFS files in parallel across large dataset using a large number of nodes (computers)[8]. The input data to be computed were so large that require a distributed processing over hundreds or thousands of nodes known as clusters. In this paper, reduce side join on more two data sets features multi-way joins is used [9] .The pseudo code of the MapReduce programming is shown in Fig. 4.

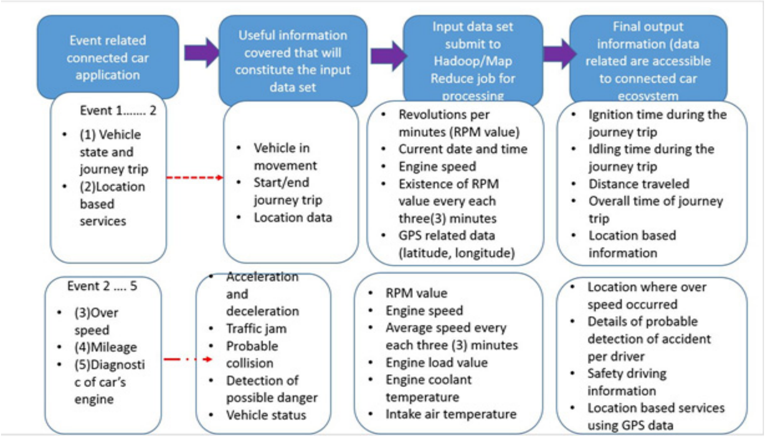
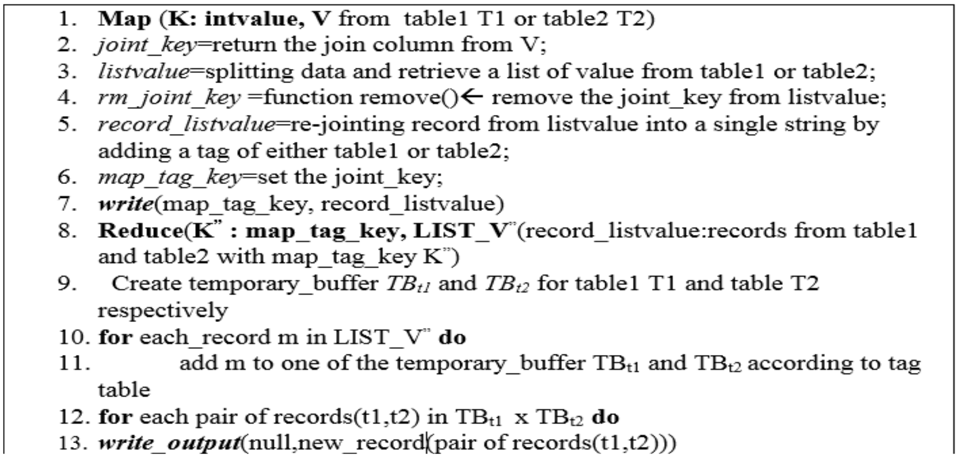

Event related connected car application
Useful information covered that will constitute the input data set
Input data set submit to Hadoop/Map Reduce job for processing
Final output information (data related are accessible to connected car ecosystem
Event 1....... 2
• (1) Vehicle state and journey trip
• (2)Location based services
• Vehicle in movement
• Start/end journey trip
• Location data
• Revolutions per minutes (RPM value)
• Current date and time
• Engine speed
• Existence of RPM value every each three(3) minutes
• GPS related data (latitude, longitude)
• Ignition time during the journey trip
• Idling time during the journey trip
• Distance traveled
• Overall time of journey trip
• Location based information
Event 2 5
• (3)Over speed
• (4)Mileage
• (5)Diagnostic of car's engine
• Acceleration and deceleration
• Traffic jam
• Probable collision
• Detection of possible danger
• Vehicle status
• RPM value
• Engine speed
• Average speed every each three (3) minutes
• Engine load value
• Engine coolant temperature
• Intake air temperature
• Location where over speed occurred
• Details of probable detection of accident per driver
• Safety driving information
• Location based services using GPS data

Fig. 3. Design of Monitoring and Analytics framework of Vehicle's data for based on Big Data technology for implementing Remote Vehicle Diagnostics services



```
Map (K: intvalue, V from  table1 T1 or table2 T2)
joint_key=return the join column from V;
listvalue=splitting data and retrieve a list of value from table1 or table2;
rm_joint_key =function remove()← remove the joint_key from listvalue;
record_listvalue=re-jointing record from listvalue into a single string by
adding a tag of either table1 or table2;
map_tag_key=set the joint_key;
write(map_tag_key, record_listvalue)
Reduce(K" : map_tag_key, LIST_V"(record_listvalue:records from table1
and table2 with map_tag_key K")
  Create temporary_buffer $TB_{t1}$ and $TB_{t2}$ for table1 T1 and table T2
respectively
for each_record m in LIST_V" do
          add m to one of the temporary_buffer $TB_{t1}$ and $TB_{t2}$ according to tag
table
for each pair of records(t1,t2) in $TB_{t1}$  x $TB_{t2}$ do
write_output(null,new_record(pair of records(t1,t2)))
```

Fig. 4. The pseudo code of the MapReduce programming for implementing Reduce side join on more two data set features multi-way join

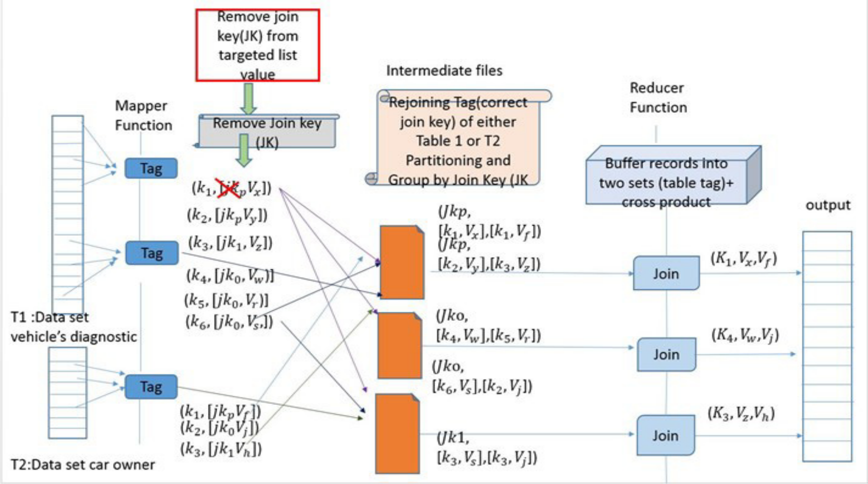

Remove join key(JK) from targeted list value
Intermediate files
Reducer Function
Mapper Function
Remove Join key (JK)
Rejoining Tag(correct join key) of either Table 1 or T2 Partitioning and Group by Join Key (JK
Buffer records into two sets (table tag)+ cross product
output
Tag
Tag
Tag
$(k_1, \cancel{[jk_p} V_x])$
$(k_2, [jk_p V_y])$
$(k_3, [jk_1, V_z])$
$(k_4, [jk_0, V_w])$
$(k_5, [jk_0, V_r)]$
$(k_6, [jk_0, V_s])$
$(Jkp, [k_1, V_x],[k_1, V_f])$
$(Jkp, [k_2, V_y],[k_3, V_z])$
$(Jko, [k_4, V_w],[k_5, V_r])$
$(Jko, [k_6, V_s],[k_2, V_j])$
$(Jk1, [k_3, V_s],[k_3, V_j])$
Join
Join
Join
(K_1, V_x, V_f)
(K_4, V_w, V_j)
(K_3, V_z, V_h)
T1 :Data set vehicle's diagnostic
$(k_1, [jk_p V_f])$
$(k_2, [jk_0 V_j])$
$(k_3, [jk_1 V_h])$
T2:Data set car owner

Fig. 5. Reduce side join on more two data set features multi-way join

Fig. 5.explains a multi-way join that features reduce-side cascade join which basically is an extension of the two-way joins within Reduce-Side join functions. A list of data sets (tables) on which we want to process is passed as part of the job configuration for allowing the Mapper and Reducer to know how many data sets to expect and what tags are related to each table. The Mapper reads tables, then points tags out from the job configuration and invokes the map function. The join key is removed from list value in order to form a list which includes all the tuples from all the data sets, then rejoining after with the tag of either table T_1 or T_2 or T_n in a single list value .The tuples based on the dataset they originate from are then proceed by the map function. After Partitioner and the Grouping function have partitioned and grouped the tuples from the all data sets by taking in consideration just the key, and ignoring the tag, the Reduce function gets the tuples sorted on the (key, tag) composite key. All tuples having the same value for the join key are perceived by the same Reducer and only one Reducer function is invoked for one key value. During execution of Reducer function, the Reducer creates buffers to hold all but the grouped list value within their composite key dynamically. Once the tuples for a particular key are divided as per their parent datasets, it is a cartesian product of these tuples to be performed. Finally, the joined tuples are written to the output.

2.2 Statical Analysis System (SAS) programming

SAS programs consist of an initial DATA step that creates the data set to be analyzed[10].The general form of a data step with the data entered is described in Fig.6

```
DATA filename;
INPUT variable_1 variable_2 … variable_n;
SAS statement_1;
CARDS;
data row 1 data row 2 . . . data row n ;
RUN;
```

Fig. 6. Pseudo code of general form of data step with data emitted in SAS programming.

3. Implementation and its results

In this paper, we develop a Remote On-line Vehicle Diagnostics application based on android. It will be used by the vehicle owner via his mobile device based android. We set up the experiment environment constitutes of a Hadoop multi-node cluster on a distributed environment using three systems (one master and two slaves, each of them is a core i5-6600Processor within 3.90GHz, 16 GB of RAM). Meanwhile, Diagnostic Trouble code (DTC) are collected into NoSQL database (MongoDB)[11], and then dumped directly into HDFS on the Hadoop cluster.

In order to compare the efficiency of Hadoop MapReduce computing using join algorithm that features reduce side join on more two data set features multi-way join against HiveQL[12] and the traditional statistical analysis, we compute the same statistical values (mean) on Diagnostic Trouble codes stored onto HDFS. Fig. 7. shows the processing time of three methods.

The Fig. 7. comes out with the following observations:

- Comparing to traditional statistical analysis system, Hadoop distributed parallel computing enhances processing speed when the size of dataset to be processed is getting bigger and bigger and not meaningfully different over a lower volume of dataset.

- Comparing data join MapReduce algorithm to HiveQL, relational data join patterns in MapReduce increases computing speed over HiveQL but it takes time to implement. HiveQL is arguably one of the tools for developers and analysts with strong Structured Query Language (SQL) skills but SQL is not suitable for every big data problem.

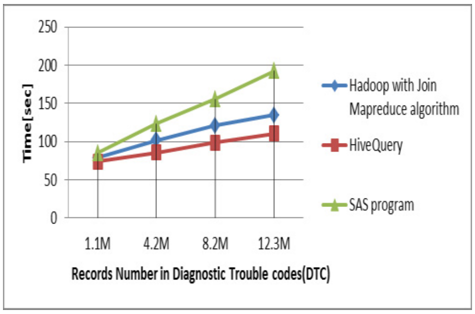


Fig. 7. Execution time of MapReduce join against SAS, HIVEQL

4. Conclusion and Future work

With the development of Hadoop platform project, now is possible to build big data solution using open source projects integrated with Hadoop. In this paper, we design a monitoring and analytics framework based on Big Data technology for processing data from vehicles using Hadoop and making final results available, allows accessing useful information via web services to the third party such as car manufacturer, transportation, emergency services has been conducted on a multi node cluster.

We compare the performance of join algorithm implemented using MapReduce programming model, Apache hive with SAS program. Future work will center on implementing and evaluating performance of Remote Vehicle Diagnostics service and specific Connected Car application on cloud platforms for instance EC2 (Amazon Elastic compute Cloud).

Acknowledgements

This work was supported by the Brain Busan 21 Project in 2016 and Nurimaru R&BD project (Busan IT Industry Promotion Agency) in 2016.

References

1. Johanson M, Dahle P, Soderberg A. Remote Vehicle Diagnostics over the Internet using the DoIP Protocol", Proceedings in The sixth *International Conference on Systems and Networks Communications. ICSNC* 2011, p226-231.
2. Kimley-Horn and Associates Inc, "Traffic Management Centers in a connected vehicle environment". *Future of TMCs in a connected vehicle*, final report, 2013, p1-27
3. Michigan Department of Transportation and Center for automotive research, "Connected Vehicle Technology Industry Delphi Study". Online at: http://www.cargroup.org/assets/files/mdot/mdot_industry_delphi.pdf, 2012, p1-22.
4. Customer Analytics.The art of possibility: connected vehicles and big data analytics. Online at: http://blogs.sas.com/content/customeranalytics/2014/12/29/the-art-of-possibility-connected-vehicles-and-big-dataanalytics/, 2014, pp.1-10.
5. Cui B, Mei H, Chin B.O. Big data: the driver for innovation in databases, *National Science Review*, Vol.1, No1, 2014; p27-30
6. Apache Sqoop, Available: http://sqoop.apache.org/
7. Tom White, "Hadoop: The definitive Guide, Third Edition", page, 411-412
8. Dean J, Ghemawat S .MapReduce: simplified data processing on large clusters. *Commun ACM*., 2008; Vol.51, p107-208
9. Jiang D, Tung AKH, Chen G. MAP-JOIN-REDUCE: Towards Scalable and Efficient Data Analysis on Large Clusters. *IEEE transactions on knowledge and data engineering*, 2011, p1299-1311.
10. M., David, "SASReduce-An implementation of MapReduce in BASE/SAS". Whitehound Limited, UK. Online at http://support.sas.com/resources/papers/proceedings14/1507-2014.pdf, pp.1-16, 2014.
11. NoSQL databases Explained. https://www.mongodb.com/nosql-explained
12. Apache Hive.Hive QL Reference. https://docs.treasuredata.com/categories/hive