Supporting Large Scale Connected Vehicle Data Analysis using Hive

Weijia Xu[1], Natalia Ruiz Juri[2], Amit Gupta[1], Amanda Deering[2],
Chandra Bhat[2], James Kuhr[2] and Jackson Archer[2]
[1]Texas Advanced Computing Center, University of Texas at Austin
[2] Center for Transportation Research, University of Texas at Austin
Email: xwj@tacc.utexas.edu, nruizjuri@mail.utexas.edu, agupta@tacc.utexas.edu, {amdeering, jkuhr, bhat, jacksonarcher}@utexas.edu

***Abstract*—Connected vehicles (CVs) are vehicles that can exchange messages containing location and other safety-related information with other vehicles, and with devices affixed to roadside infrastructure. While the main purpose of vehicle connectivity is to enhance safety, the data generated by CVs has an enormous potential to support transportation planning and operations. However, handling the vast volume of data produced by CVs presents considerable challenges for researchers in the transportation domain. This paper presents a case study of using HIVE to facilitate CV data analysis based on the largest CV data set publicly released to date. We characterize the data analytic tasks that are expected to enable transportation planning research, and investigate several approaches to increase the corresponding query efficiency and throughput. This study compares the use of HIVE in conjunction with the MapReduce and Spark programming frameworks, analyzes its performance using different data storage formats, and exemplifies potential use cases.**

Keywords: Connected Vehicles Data, Big Data, Transportation Planning, Spark, Hive

I. INTRODUCTION

Connected vehicles (CVs), including vehicles capable of "Vehicle to Vehicle" (V2V) and "Vehicle to Infrastructure" (V2I) communications, may become a reality in the next 10 years [1]. By sending and receiving short communications, known as Basic Safety Messages (BSMs), CVs receive key information about their surroundings that can be relayed to the driver to aid safe operations. CVs are projected to make the roadways safer through real time information exchange [1]. While much of the focus and reasoning for implementation of V2V/V2I technologies remains on the safety benefits to the driver, the potential for data collection and analysis to inform real time traffic operations and mobility planning is also remarkable. However, the volume and complexity of CV data are also tremendous and present a big data challenge for the transportation research community.

A one year long pilot project conducted by the U.S. Department of Transportation in 2013 generated hundreds of gigabytes of CV data. Two months of sanitized mobility data from this study have been publicly released. Despite the relatively small scale of the study, the released data already exceeds the capability of most of the data analysis tools typically used by transportation researchers and practitioners today. As the complexity and computational resources requirements are expected to grow at least linearly with the number of deployed vehicles and over the analyzed time period, datasets from large-scale deployments may be multiple orders of magnitude larger than the pilot datasets. Therefore, new technology and analysis frameworks must be investigated and developed to enable research centered on large CV deployment test beds.

In this paper, we report and compare our efforts to develop scalable solutions for CV data analysis using Hadoop clusters [2] [3] and Spark [4] through Hive [5]. A set of basic analysis tasks to support transportation planning applications are identified and mapped to an appropriate workflow in Hive. This work characterizes the workload of the proposed tasks, evaluates the scalability of using Hive to support them, and compares the performance of the MapReduce (Hadoop) [6] and Spark programming frameworks.

Hive is an open-source data warehouse application that supports distributed queries over a Hadoop cluster. It can utilize both, Hadoop's MapReduce and Spark to support efficient data summarization and query over large-scale data stored in a Hadoop Distributed File System (hdfs) [7]. Hive provides a SQL like interface, HiveQL, for querying and analyzing data. This feature is especially valuable for transportation researchers whose traditional workflow uses database such as PostgreSQL. Furthermore, an open source spatial analysis extension is also available to support a subset of features similar to those in PostGIS [8]. The performance of Hive is affected by hardware specific parameters and two factors that are often specific to the workload: the storage format used to backup the query operations, and the query execution engine. HiveQL also has limitations when handling more complicated query requirements, such as join queries and user defined functions (UDFs). Since the optimal choice of appropriate values for such options is often workload dependent, one of goals of this study is to characterize the workloads expected from large scale CV data analyses, and to understand current challenges and limitations.

Our work explores the use of real world connected vehicle data from the most comprehensive data set released to date: the Safety Pilot Model Deployment (SPMD) data. The study, conducted in Ann Arbor, Michigan, involved over 2,700 vehicles, and provides an opportunity to better

978-1-4673-9005-7/16/$31.00 ©2016 IEEE

understand the characteristics of collecting, processing and analyzing real world, large scale CV data [9].

Our contributions include three real world use cases of CV data for transportation planning, SQL level solutions to support those analysis and lessons learned through our performance evaluation. Performance evaluation results indicate that Hive can provide much needed scalability for a subset of basic data pre-processing tasks. In comparison, supports for geospatial-based analyses, which are central to transportation planning, seem to be less scalable. Therefore, the implementation of new operators emerges as an important requirement to support more scalable data analyses. Our results also indicate trade-offs between different types of query engine and storage formats.

The rest of the paper is organized as follows: Section II provides an overview of CV data, including its characteristics, and the applications that have been proposed in the literature, and a review of big data processing systems. Section III describes the SPMD dataset and the proposed CV data processing workflow and corresponding requirements. Section IV details the testing environment and comparison results. We conclude and describe future work in Section V.

II. Background and Related Work

A. Connected vehicle data

CV communications rely on the broadcast of Basic Safety Messages from individual vehicles within a Vehicular Ad-hoc Network (VANET) [10]. The Society of Automotive Engineers (SAE) has established communication standards for connected vehicles (compiled in SAE J2735). These standards define the key aspects of the Dedicated Short Range Communication (DSRC) for V2V and V2I (collectively, V2X) technologies, such as the format of BSMs, channels and frequencies, and beaconing intervals [11]. To broadcast messages through the VANET, CVs require specialized equipment typically referred to as Onboard Units (OBUs). The first commercial vehicles to have OBUs installed are expected in summer 2016 from Cadillac [12].

There are two types of BSMs: BSM I messages are centered on the communication of safety-related information, including vehicle size, position, speed and heading; BSM II messages contain additional, non-safety-critical information, such as brake status, lights status, rain sensor, and whether or not systems like antilock brakes and traction control are active. CVs broadcast BSM I and BSM II (when warranted) at an adjustable rate of up to 10 messages per second. The information communicated across the VANET can be supplemented with data broadcasted by the infrastructure (V2I), including Signal Phasing and Timing ("Spats"), intersection geometry ("MAP Data"), position correction, and traveler information messages (TIMs).

The US Department of Transportation (USDOT) has recommended a slew of potential planning and operations applications that could be improved through the emergence of Connected and Autonomous Vehicles [1]. Among these are improvements to intelligent traffic signal systems, integrated dynamic transit operations, integrated corridor management, fleet management systems, and even streamlined international border crossings. In academia, researchers are actively studying the potential impacts of CV data on traffic operations [13], and to a lesser extent, on transportation planning [14]. Additionally, the Michigan Department of Transportation commissioned a study, completed in 2012, to begin building the tools to process and analyze connected vehicle data [15]. However, due to a lack of real world large scale deployments until recently, much of the existing research had no option but to rely on simulated data or relatively small real world samples. These sample sets, while very beneficial, were unable to empirically measure the challenges and questions that emerge when working with data from a large scale, real world deployment.

B. Spatial data analysis with Hive

Hive was first introduced in 2009 as a data warehouse solution utilizing a Hadoop cluster [6]. Hive supports multiple types of data sources and file formats, including structured text format, such as CSV and JSON, and serialization format, such as Avro, Parquet and ORC. Each of the supported data sources can be ingested in Hive as one or more tables. Hive provides a SQL like query language, known as HiveQL, for users to search over tables in the system. SQL requests are transformed into one of more distributed computation tasks through built-in query engines. Two Hive query engines are investigated in this study: MapReduce programming model [6] and Spark programming model [4].

Although Hive has been successfully used in industry for business analytics, and in many domains for scientific discoveries, the application of Hive in transportation is still limited. *Hadoop-GIS* is a data warehouse framework proposed to support general spatial analysis [16]. *Hadoop-GIS* includes MapReduce implementations of several common spatial analysis tasks such as spatial join, spatial containment testing and spatial filters. However, unlike Hive, *Hadoop-GIS* does not support query language. A user can use *Hadoop-GIS* as a programming library or through command line interface. *Spatial hadoop* is a similar research project that utilizes a MapReduce programming framework for spatial analysis [17]. It includes a storage layer for efficient data management and an operation layer to perform set operations such as range query and spatial join. *Spatial hadoop* has a simple query language specification to query spatial data. Both projects are implemented only in a MapReduce framework and use Hadoop clusters, and have limited interoperability with non-spatial data. *Spatial framework for Hadoop* is an open source java library that supports spatial analyses using Hadoop [18]. This framework does not directly provide a data warehouse environment, but focuses on implementing basic spatial analysis functions. *Spatial framework for Hadoop* provides UDF extensions that can be used directly with Hive. The integration with Hive

provides users more flexibility to query both spatial and non-spatial data, and the opportunity to take advantage other programming frameworks and storage formats. We choose to use *Spatial framework for Hadoop* in our testing.

III. Data Summary And Analysis Requirment

A. Connected Vehicle Data

The Connected Vehicle Safety Pilot Model Deployment (SPMD) [9] is a model deployment of connected vehicles conducted by The University of Michigan Transportation Research Institute (UMTRI) and the National Highway Traffic Safety Administration (NHTSA) between 2012 and 2013. Its purpose was to test the effectiveness of connected vehicle safety applications, as well as to evaluate the feasibility of the DSRC technology. Two months of data collected through the SPMD are freely available on the USDOT Research Data Exchange (RSE) website [19].

Approximately 2,700 cars, trucks, and transit buses participated in this study, with some of those vehicles being equipped with more comprehensive CV technology than others:

- 64 cars and 3 trucks were "fully integrated," which means that they had integrated safety systems (ISS) installed during vehicle production. The ISS broadcasts and receives BSMs, and generates warnings to the driver.
- 300 cars were equipped with aftermarket safety devices (ASD), which are installed after initial vehicle manufacturing. These vehicles can send and receives BSMs as well as generate warnings to the driver, and they have approximately the same capabilities as vehicles with an ISS.
- 16 trucks and 3 buses were equipped with retrofitted safety devices (RSD), which are installed on a vehicle after the vehicle has completed the manufacturing process. These devices are specifically being developed for transit buses and trucks, and include a driver interface in such way that they can broadcast and receive BSMs, and generate warnings to the driver.
- 2,305 cars, 60 trucks, and 86 buses were equipped with a vehicle awareness device (VAD). These vehicles represented the vast majority of the vehicles in the Safety Pilot. The VAD is only capable of sending BSMs over DSRC, but not receiving them. The VAD does not generate warnings to the driver of the vehicle, but it transmits the vehicle's speed and location.

Road side units (RSUs) were also part of this study, some of which logged received BSMs and relayed this information to the appropriate servers via network/fiber connections. Most of the data were logged onboard vehicles and downloaded at a later time. The data sets used in this paper were obtained from vehicles equipped with different data acquisition systems. Both include vehicle position and motion data. For additional details of those data sets and other available CV data, please refer to [20].

B. Requirements of CV Data Analysis for transprotation research

The research team has identified several potential applications of CV data to enhance transportation planning methodologies. These include improvements to the generation of model inputs, the validation of model results, and ultimately the refinement of modeling assumptions. Transportation planning models are helpful to understand the characteristics and magnitude of future travel demand, and its impact on experienced travel conditions. These models are used by planning agencies to comprehensively evaluate the potential effect of a variety of traffic management and operation techniques, the benefits of infrastructure investments, and the value of deploying new technologies, among others. While traditional models make significant simplifying assumptions, a number of advanced modeling techniques are being increasingly used to enable more realistic analyses (e.g. Dynamic Traffic Assignment [21] and Activity Based Models [22]). The appropriate incorporation of CV into planning models, and the development of methodologies to take advantage of their data, requires significant research. The following use cases exemplify the range of potential applications of CV data. A significant portion of these use cases take advantage of the fine-grained location data generated by CVs. While such data may become available from other sources, the potential pervasiveness of CV data, combined with the availability of information that is not typically collected by positioning systems alone, makes the analysis of CV data quiet unique. Some of the proposed use cases may be applicable to the analysis of location data generated from other sources.

Use case 1: Improving travel demand estimates for traffic assignment models and real-time applications.

Traffic assignment models are used in transportation planning to understand the interaction between travel demand and the transportation system [23]. Such relationship affects the route choice of travelers under recurrent traffic conditions, which ultimately determines the overall system performance. Typical inputs include travel demand estimates and transportation network characteristics. Advanced models usually specify travel demand by time of day. Depending on the modeling framework, demand may be described by a time-dependent origin-destination matrix among transportation analysis zones (TAZs), or provided as a set of individual "tours" that encompass activities throughout the day. The analysis of CV data can lead to a better understanding of trip departures/arrival patterns by time of day. Such information may be used to refine time-dependent origin-destination matrices, validating the process used to generate them. If sufficient data is available, CV data may be used to generate travel demand estimates for a region, potentially in real time. The latter can drastically improve the planning of incident response strategies, such as traffic diversions. Travel demand analysis often requires mapping trip origins

and destinations to a set of user defined "zones" that facilitate the interpretation of modeling results. Additionally, temporal aggregation is usually necessary to describe time-dependent trip-making behavior. Zones may be approximated using an arbitrary grid of appropriate size for preliminary and qualitative analyses (Figure 1).

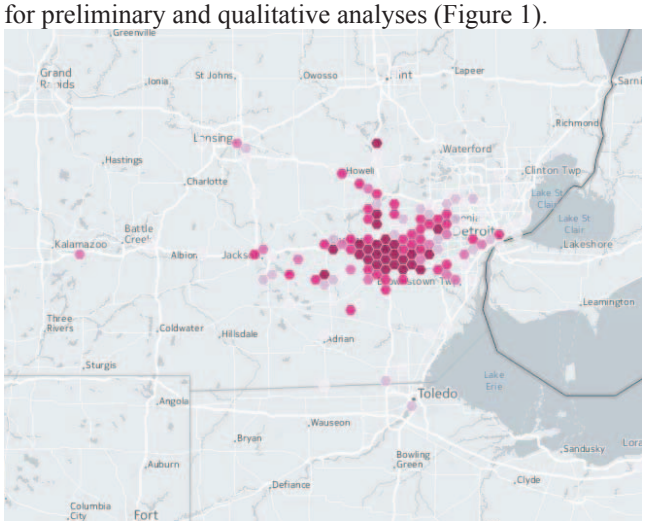

Figure-1 Spatial distribution of trip origins for selected departure time range, using small hexagonal grid.

Use case 2: Validating the assumptions underlying route choice in traffic assignment

Traffic assignment models usually assume self-interested travelers that distribute themselves throughout the network by choosing paths with minimum travel costs. All used paths are expected to have equal and minimum travel times. While such behavior, also known as equilibrium, is widely accepted to reflect recurrent traffic conditions, more complex models may be desirable as real-time information becomes more pervasive. CV data may be used to understand daily route choice patterns and compare them with the output of the theoretical models. Further, CV data combined with historical system data may lead to a better understanding of the impact of additional route attributes, such as safety and reliability, into the route choice process. The type of analysis proposed in this section requires studying the trajectory data of trips with similar origins and destinations, and comparing it to theoretically optimal paths (Figure 2).

Use case 3: Validating estimates of traffic conditions

The ultimate goal of most transportation models is to provide estimates of traffic conditions under various scenarios for some (or all) roadway segments. CV data may be used to validate the outcomes of such models at almost any desired temporal and spatial aggregation level, and refine their assumptions. Such analyses are valuable for both, models that use traffic flow theory to estimate the propagation of traffic through the network, and for statistical, data-driven models. The analysis of roadway performance requires appropriate spatio-temporal aggregation. Qualitative analyses may be conducted by visualizing aggregate metrics, while modeling efforts are likely to require the definition of an appropriate representation system for the roadway network, an the mapping of the observed properties to such system.

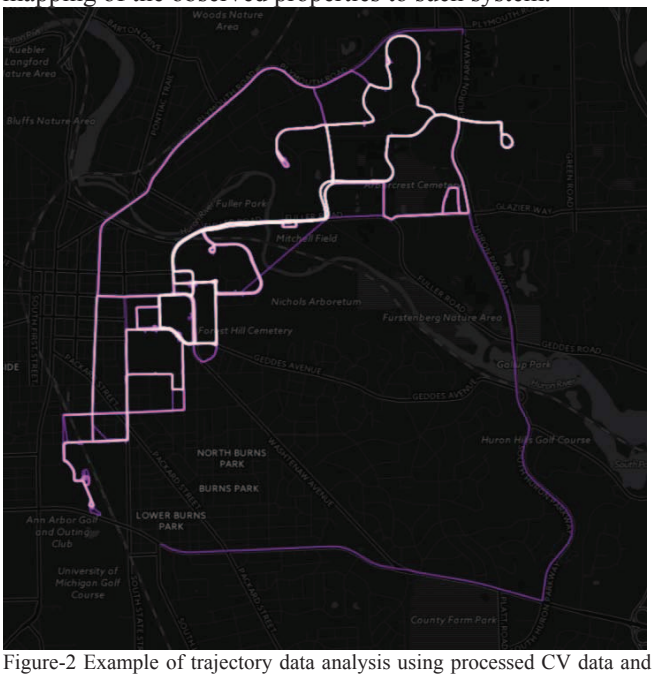

Figure-2 Example of trajectory data analysis using processed CV data and open-source GIS software.

C. Mapping use case reuqirements to solutions using Hive

We identify and categorize four types of queries that are needed to support three potential use cases listed above.

1) Type I: Query using group-by aggregations

The CV data includes multiple high-frequency sensory outputs such as precise location information of the vehicle, speed and direction every tenth of a second. Therefore, one trip can consist of thousands of data points. For example, table *hv_primary* has 83,384,194 rows describing 14,436 trips. Aggregation is required in order to extract meaningful information for data analysis. Based on the complexity of the aggregation, there are three groups of queries.

```
Query example 1
SELECT
  deviceid, trip, AVG(gps_speed)
FROM
  hv_primary
GROUP BY
  trip, deviceid;
```

Query 1 is a simple aggregation query to retrieve the average value of the speed during a trip for each vehicle. This query only involves three columns and one aggregation function. For other analyses, more columns and aggregation functions may be required.

Query 2 includes four aggregation functions to find the duration of each trip, the length of time during which breaks were in use, and the number of lane shifts. For table

hv_primary, a simple aggregation of all attributes by trip and vehicle involves 30 columns.

```
Query example 2
SELECT
  deviceid, trip, AVG(gps_speed) as speed,
  MAX(time) as duration,
  SUM(invehicle_brake_status) as brake,
  SUM(lanetrack_shift_successful) as shift
FROM
  hv_primary
GROUP BY
  trip, deviceid;
```

2) Type II: Aggregation over windowing

The second type of query requires further selection within groups and cannot be accomplished with simple built-in functions. For example, the following query provides the end location of each trip:

```
Query example 3
SELECT
  deviceid, trip, time, gps_latitude,
  gps_longitude
FROM(
   SELECT
      *, ROW_NUMBER() OVER
         (PARTITION BY
           trip order by time desc) as rn
   FROM
      hv_primary
) sub1
WHERE
   rn=1;
```

Query 3 can retrieve the last record from each trip and report its location, which allows computing the duration of the trip. Practical applications may require identifying trip origin and destination, which involves multiple aggregations over windowing (Query 4).

```
Query example 4
SELECT
  deviceid, trip, time,
  FIRST_VALUE(gps_latitude)over w as o_lat,
  FIRST_VALUE(gps_longitude)over w as o_long,
  LAST_VALUE(gps_latitude) over w as d_lat,
  LAST_VALUE(gps_longitude) over w as d_long
FROM
  hv_primary
WINDOW w AS
  (PARTITION BY trip order by time);
```

3) Type III: Trajectory data aggegation and analysis

Vehicle trajectories are the paths followed by vehicles from origin to destination in any given trip. Trajectories are defined by all the points at which a CV broadcasts location and other information, and can be used to validate and enhance transportation models. Some analyses may be accomplished by considering only the location of trip origins and destinations, which may be grouped by proximity and used in qualitative analyses, or mapped to an appropriate representation system for modeling.

The queries proposed in this section are designed for analyses that require considering many, if not all, points in a trajectory. The latter include the computation of distance traveled and the visualization of the selected route. Such computations often require additional assumptions concerning the geo-spatial projection system to produce an accurate calculation.

To support the analysis of trajectories, we propose the use of the existing spatial analysis framework for hadoop [18]. Query 5 shows an example of computing distances travelled for all trips.

```
Query example 5
SELECT
  deviceid, trip,
  ST_GeoLength(ST_SetSRID(
       ST_LineString(
         collect_list(point)),4326))
  as distance
FROM (
  SELECT
   deviceid, trip,
   ST_Point(gps_longitude, gps_latitude)
   as point
   FROM hv_primary
)sub
GROUP BY
   deviceid, trip;
```

In Query 5, *ST_GeoLength, ST_SetSRID, STLineString, ST_point* are four user defined functions using existing implementations in the spatial analysis framework for hadoop. The query generates a geospatial point object based on the *gps_longitude* and *gps_latitude* values for each selected record. All points from the same trip are then aggregated and used to compute the total distance traveled in that trip.

4) Type IV: Geo-spatial aggregation.

CV data consists of millions of data points representing a status broadcast at a specific point in space and time. In order to analyze system performance based on such data, significant temporal and spatial aggregation is necessary. For qualitative and exploratory analyses, a grid may be defined to conduct preliminary aggregations. A grid is a set of polygons expanding a given area. In the context of this work, each polygon may be also referred to as a cell or a zone. The simplest grid consists of a set of squares of uniform size. More complex grids using hexagonal polygons may also be used to better suit the road network and geographic features.

Mapping a point, defined by its coordinates, to a grid cell requires (in the worst case) going over all grid cells, which may be taxing in grids covering a large area, or when using small cells. Spatial filters may support efficient analyses: Query 6 exemplifies the identification of all trips originating on grid cell 100.

```
Query example 6
SELECT
  deviceid, trip, time, gps_longitude,
  gps_latitude
FROM
  grid_json, hv_primary
WHERE
  grid_json.gid = 100
AND
    ST_Intersects(grid_json.geometry,
      ST_Point(gps_longitude, gps_latitude))
AND
   time=0;
```

IV. Performance Evaluation Results

The performance evaluation has three goals: 1) to investigate the feasibility of supporting the tasks described in the previous section and identify corresponding workloads; 2) to explore the most appropriate usage model and settings of Hive; and 3) to identify additional approach to optimize and bridge the gaps.

A. Data and Testing Enviroment

Our experiments were conducted using the Rustler computing cluster at the Texas Advanced Computing Center. The Rustler cluster is a dedicated Hadoop cluster consisting of 64 computing nodes. Each node is equipped with dual 8-core Intel Xeon processors, 128GB memory and 16 TB SATA drives. The system is running Cloudera distributed Hadoop version 5.8 and Hive version 1.1.

The data used in this testing are primarily the *hv_primary* data in DAS2 dataset as described in Section 3. The *hv_primary* set includes logs of various sensory outputs installed on the vehicle, and it was registered in Hive as a table with 83,384,194 rows and 45 columns. The table for *data_wsu* includes 80,167,380 rows and 23 columns. Most of the columns contain numerical data. The data set was obtained as comma separated files with size of 27.1GB. In addition to the CV data, a hexagonal grid with 226,044 cells was generated using GIS software and ingested in JSON format. Each cell is described as a polygon and identifiable through two IDs.

In addition to the query examples given above, variations of those queries were also tested to further assess their computational requirements. For query Type I (group-by aggregation) four additional versions were with different number of columns and aggregation functions were considered. For Type II queries (using windowing), additional variations explore the impact of the total number of records. For Type III queries, the analysis was centered in the evaluation of performance with different selection conditions. For Type IV queries, variations were generated using different grids with a different number of cells. In total, our tests included 23 different queries.

B. Basic workload chacherstics study

We first measured the workload characteristics of the four types of queries described in Section A. For this study, we only used *hv_primary* table.

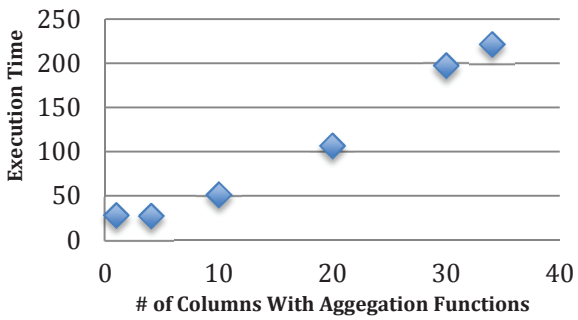

Figure-3 Execution time vs. number of columns with aggregation function for Type I queries.

Figure 3 shows that the execution time increases linearly with the number of columns on which aggregation functions are used. Figure 1 shows the outcomes of six different queries including Query 1 and 2 shown above and four other variations. Each query consists of one select statement and a varying number of columns on which aggregations were performed (1, 4, 10, 20, 30 and 34). The considered aggregation functions were all built-in, such as sum, max, min, count, avg and count distinct. This result was obtained using the CSV text file storage format and the MapReduce engine. Ninety-five mappers and one hundred reducers were used in all cases. Note that there is no significant differences in the running time between using 1 and 4 aggregation functions due to the fixed overhead of setting up the MapReduce job and reading the data.

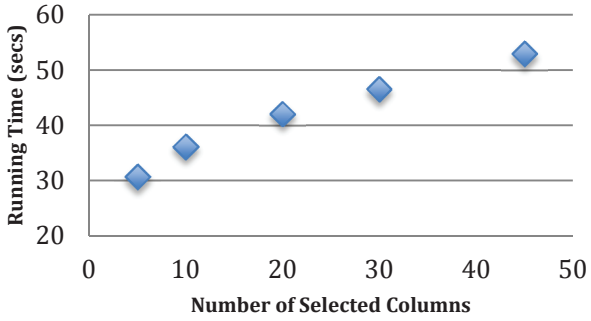

Figure-4 Running time vs. number of selected columns for Type 2 queries.

Four variations of Type 2 queries (Query 3) were tested with different number of aggregated columns. Figure 4 shows that more computational resources are required as the number of columns increases. All queries in Figure 4 involve one select statement and one window partition. Increasing the number of aggregated columns in the same query increases the execution time significantly. Figure 5 compares four variants of Query 4 with different number of aggregated columns (1, 2, and 6) over a single window.

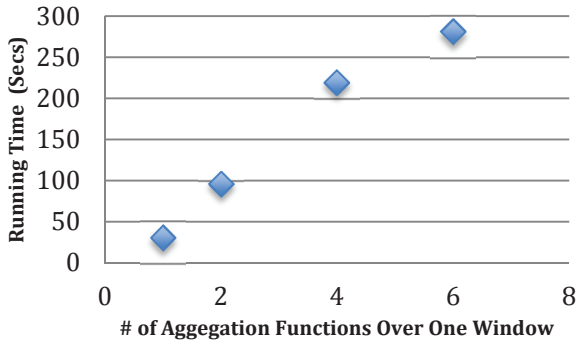

Figure-5 Running time vs. number of aggregation functions used over windowing.

Using multiple windows in the same query caused a significant increase the execution time. Our results show that an alternative version of Query 4 using two partition windows takes 4,992 seconds, in contrast to 219 seconds for the single window case. This is due to the fact that multiple windows are converted to join operations.

Type 3 queries were also observed to require a longer running time when using additional aggregation functions and UDFs. These queries seemed less sensitive than others, requiring an average of 46.834 seconds on *hv_primary*.

The performance of Type 4 queries was observed to depend on the number of geometry cells in the grid. The cost of determining the intersection between points and cells was observed to be relatively independent of the cell size. Therefore, for the same geographical area a high-resolution grid requires much more computational effort. Figure 6 shows the outcomes of seven variations of Query 6 using different number of cells.

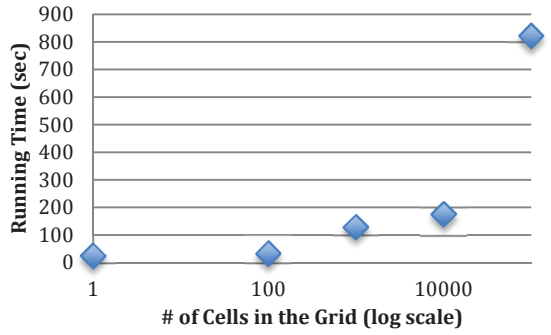

Figure-6 Workload characteristics for type IV queries.

C. *Comparisons between the MapReduce and Spark engines*

Hive can convert query statements using either MapReduce or Spark as the programming model. The results presented in the previous section were obtained using MapReduce. This section explores the performance of Spark on the same queries.

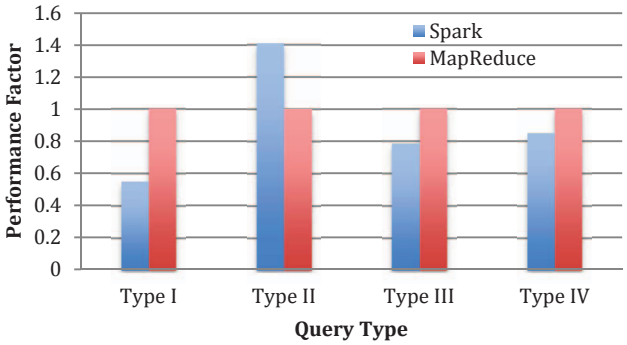

Figure-7 Normalized performance for each query type using Spark and MapReduce programming model with Hive. The y axis represents the ratio between running time using Spark and MapReduce. Smaller values indicate faster running times in Spark than in MapReduce.

Figure 7 compares both programming models. The results are normalized by the running time on the MapReduce programming model. When the ratio is smaller than one, it means Spark is faster than MapReduce. The number of executor instances of Spark was set to 95, the same as the number of Map instances used in MapReduce model. Spark performs better than MapReduce in three types of queries. For Type I, Spark run time was almost half of the MapReduce time. However, the Spark engine seems to struggle to support the windowing feature. The Type II query shown in Figure 5 only uses one window, and Spark was observed to be 40 percent slower than the MapReduce implementation. Additional queries involving more than one aggregation window took up to 10 times more time to run using Spark than Map Reduce.

D. *Impact of of storage format.*

In addition to the different query execution engines, Hive supports different data storage formats. While the original files are in text format, new tables can be created and stored using alternative formats. We tested the performance of using Spark with Parquet and ORC format.

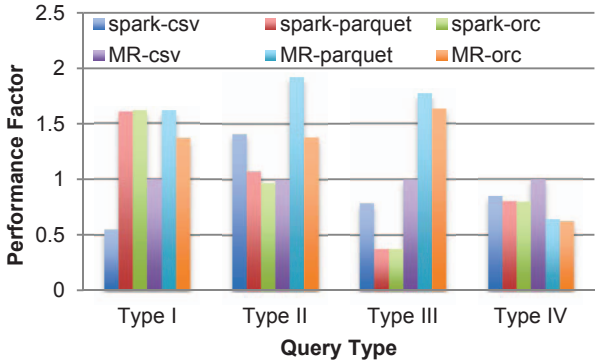

Figure-8 Comparison of different types of storage format and execution engine combinations for different types of queries. The number of data splits are different for each format on the same datasets. The perfromance factor reflects the execution time relative to using the MapReduce programming model with text format.

Figure 8 shows the performance comparison for selected queries from each category under different configurations. The results suggest that there is not a single configuration setting that outperforms others across all query types.

Both, the Parquet and ORC formats, are binary storage formats that reduce the space required to store information. However, they also introduce limitations on how the data can be split and the number of mappers and reducers that may be used. The effect of such changes is also workload dependent. The results shown in Figure 6 were computed while keeping other cluster-related settings constant. The number of data splits was reduced from 95 using text format to 26 for Parquet format, and 26 for ORC format. Smaller number of job splits can benefit queries such as Type IV, where user defined functions with limited parallelisms account for a significant part of the computation. However, it has negative effects on other types of queries. Further setting manipulation in order to produce similar levels of data split across all formats led to similar performance levels for queries Type I, II and III (and a deterioration of the performances of Type IV queries).

E. Comparison with PostgreSQL

Table-1 running time comparison between PostgreSQL and Hive

Query	Type	PostgreSQL (sec)	Hive (sec)	Speed up
1	I	425.71	28.33	15.03
2	I	302.95	27.75	10.92
3	II	307.64	30.615	10.05
5	III	3509.00	44.172	79.44
6	IV	24.70	25.862	0.95

Table-1 shows comparison on running time for example queries using PostgreSQL (version 8.1.23) and Hive (version 1.1). The PostgreSQL queries were executed on the database server (Intel Xeon x5677 with 16 (3.47Ghz) hardware threads and 98G of memory) at CTR which is regularly used for similar analysis tasks. However, query 4 cannot be finished within a few hours using PostgreSQL. The running time of other queries are shown in Table-1. The running time of HiveQL are the running time using Map Reduce query engine. We computed the speed up as

$$\text{speed up} = \frac{\text{Running Time of PostgreSQL}}{\text{Running Time of Hive}}$$

A higher speed up value indicates a greater speed up. The highest speed up, 79.44, comes from running type III query. For type I and type II query, using hive can give over 10 times faster than using PostgreSQL. For type IV query, the running time are similar for PostgreSQL and Hive which indicates the current approach does not take much advantage for the distributed processing capability of Hive.

V. CONCLUSION AND ONGOING WORK

In this paper, we investigated the feasibility of using Hive as the main data storage manager and analysis platform for connected vehicle data analysis. We mapped analysis tasks expected to be required for three common use cases to four types of queries. Our assumption is that more sophisticated analyses can be done using a combination of such queries. Identifying and profiling the selected queries is important in both, providing basic building blocks to enable domain-specific analyses and facilitating further optimization to improve its performance and usability.

Our results indicate that using Hive for CV data analysis is a viable and scalable approach. Most of the queries exemplified in this work may take minutes to hours to complete in a traditional PostgreSQL database. At the same time, our results also show very different workload characteristics interweaving both data intensive and computation intensive aspects. This makes identifying a set of optimal setting that works best across all use cases a challenging task. Nevertheless, the following general observations can be made based on the work presented in this paper:

1) The Spark processing engine generally achieves better performances than the MapReduce programming framework. However, our tests suggest that Spark does not support the windowing feature adequately, which may require further investigation and improvements.

2) The use of a binary format for data storage has the benefit of reducing the size of the data, but it also changes the number of data splits and affects the overall workload balance. As a result, the overall effect is mixed across analyzed tasks.

3) While Hive is a general application for distributed data warehousing and querying, several essential features required to support CV data analysis must be implemented as extensions of user defined functions. The performance of user-defined functions can become a bottleneck, as shown in Type IV queries, as it reduces the parallelisms.

Based on these observations, part of our ongoing work will be implementing spatial join operators that can be used to make Type II and Type IV queries in Hive more efficient.

Preliminary analyses of CV data suggest that such data has the potential to support the enhancement of transportation planning methodologies by contributing to the generation of model inputs, the validation of model results, and ultimately the refinement of modeling assumptions. Precise location data collected over time, once appropriately aggregated and analyzed, can be used to understand travel patterns by time of day, which is a basic input to transportation planning models. The analysis of vehicle trajectories can support a more thorough understanding of route-choice behavior, and validate or improve the behavioral assumptions that underlie existing models. The spatial analysis of vehicle-based data such as vehicle speed and breaking can enhance the understanding of system performance, safety, and how these affect travel patterns and route choice. While the analysis presented in this work uses the DAS dataset, a similar approach can be used to analyze the BSM data. Further, the methodologies proposed in this paper are appropriate for other sources of precise geo-located data.

ACKNOWLEDGMENT

This work has been partially funded through D-STOP, a US DOT Tier 1 University Transportation Center, and

motivated by ongoing work for the North Central Council of Governments (NCTCOG). The work is also funded in part by NSF XSEDE project and also supported by NSF award #1341711 for the data intensive computing resource Wrangler at Texas Advanced Computing Center.

REFERENCES

[1] J. Harding et al., "Vehicle-to-Vehicle Communications: Readiness of V2V Technology for Application.," National Highway Traffic Safety Administration, Washington, DC, DOT HS 812 014, 2014.

[2] Apache Foundation. (2007) Apache Hadoop Project. [Online]. http://hadoop.apache.org/

[3] Tom White, *Hadoop: The definitive guide*.: O'Reilly Media, 2012.

[4] Matei Zaharia et al., "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, 2012, p. 2.

[5] Ashish Thusoo et al., "Hive: a warehousing solution over a map-reduce framework," in *Proceedings of the VLDB Endowment*, vol. 2, 2009, pp. 1626-1629.

[6] Jeffrey Dean and Sanjay Ghemawat, "MapReduce: a flexible data processing tool," *Communications of the ACM*, vol. 53, no. 1, pp. 72-77, 2010.

[7] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, and Robert Chansler, "The hadoop distributed file system," in *Proceedins of IEEE 26th symposium on mass storage systems and technologies (MSST)*, 2010, pp. 1-10.

[8] PostGIS refernece guide. [Online]. http://postgis.net/docs/reference.html

[9] D. Bezzina and J. Sayer, "Safety Pilot Model Deployment: Test Conductor Team Report," National Highway Traffic Safety Administration, Washington, DC, DOT HS 812 171, 2015.

[10] F. Jiménez, "Connected Vehicles, V2V Communications, and VANET," *Electronics*, vol. 4, no. 3, pp. 538-540, 2015.

[11] Society of Automotive Engineers, "Dedicated Short Range Communications (DSRC) Message Set Dictionary," J2735 Ground Vehicle Standard, J2735, 2016.

[12] General Motors. Cadillac to Introduce Advanced “Intelligent and Connected” Vehicle Technologies on Select 2017 Models. [Online]. https://media.gm.com/media/us/en/gm/home.detail.html/content/Pages/news/us/en/2014/Sep/0907-its-overview.html.

[13] D. Fajardo, T.-C. Au, S. Waller, P. Stone, and D. Yang, "Automated Intersection Control. ," *Transportation Research Record: Journal of the Transportation Research Board,* , vol. 2259, pp. 223-232.

[14] E. Dennis, Q. Hong, R. Wallace, W. Tansil, and M. Smith, "Pavement Condition Monitoring with Crowdsourced Connected Vehicle Data," *Transportation Research Record: Journal of the Transportation Research Board*, vol. 2460, pp. 31-38, 2014.

[15] L. T. Mixon, J. K. Garrett, and B. C. Krueger, "Vehicle Infrastructure Integration (VII) Data Use Analysis and Processing," Michigan Department of Transportation, Summary Report 2012.

[16] Ablimit Aji et al., "Hadoop GIS: a high performance spatial data warehousing system over mapreduce," *VLDB Endow.*, vol. 6, no. 11, 2013.

[17] Ahmed Eldawy and Mohamed F. Mokbel, "Spatialhadoop: A mapreduce framework for spatial data," in *2015 IEEE 31st International Conference on Data Engineering*, 2015, pp. 1352-1363.

[18] Esri. (2016, Oct) Spatial Framework for Hadoop. [Online]. https://github.com/Esri/spatial-framework-for-hadoop

[19] Research Data Exchange. [Online]. https://www.its-rde.net/

[20] Booz Allen Hamilton, "Safety Pilot Model Deployment - Sample Data Environment Data Handbook," U.S. Department of Transportation Intelligent Transportation Systems Joint Program Office, 2015.

[21] Y.-C. Chiu et al., "Dynamic traffic assignment: A primer," *Transportation Research E-Circular*, no. E-C153, 2011.

[22] Chandra R. Bhat and Frank S. Koppelman, "Activity-based modeling of travel demand," in *Handbook of transportation Science*.: Springer US, 1999, pp. 35-61.

[23] Yosef Sheffi, *Urban transportation network*.: Pretince Hall , 1985.