Use of Big Data Technology in Vehicular Ad-hoc Networks

Conference Paper · September 2014

DOI: 10.1109/ICACCI.2014.6968352

2 authors:

Punam Bedi
University of Delhi
207 PUBLICATIONS **1,214** CITATIONS
SEE PROFILE

Vinita Jindal
Keshav Mahavidyalaya, University of Delhi, Delhi, India
17 PUBLICATIONS **54** CITATIONS
SEE PROFILE

Use of Big Data Technology in Vehicular Ad-hoc Networks

Punam Bedi
Department of Computer Science
University of Delhi
Delhi, India
punambedi@ieee.org

Vinita Jindal
Department of Computer Science
University of Delhi
Delhi, India
vjindal@keshav.du.ac.in

Abstract— **Big Data technology is becoming ubiquitous and depicting key attention of researchers in almost all areas. VANET is a special form of MANET that uses vehicles as nodes in a network. By applying Big Data technologies to Vehicular Ad-hoc Network (VANET), one can gain useful insight from a huge amount of operational data, to improve traffic management processes such as planning, engineering and operations. VANETs access large data during the real time operations. In this paper we map VANET characteristics to Big Data attributes stated in literature. Further, we evaluate the performance of Dijkstra algorithm used for routing in vehicular networks on Hadoop Map Reduce standalone distributed framework as well as on multinode cluster with 2, 3, 4 and 5 nodes respectively. The results obtained confirm that increasing the number of nodes in Hadoop framework, processing time for the algorithm is greatly reduced.**

Keywords— Big Data; Hadoop; Map Reduce; VANETs; Dijkstra Algorithm; GSR; Shortest Path; Distributed computing.

I. Introduction

The exponential growth in terms of complexity and capacity of data in last few years has led to notable research in the field of Big Data technology. The terminology of Big Data is becoming ubiquitous now a days. Big Data technologies are still in the nascent stage of development. Big Data today, is what the web was there in 1993. The data on the web is growing and the web might get bigger and bigger, but very few of us really recognized what this “Big” meant. Today, we believe that we aren’t even abrading the surface of the Big Data opportunity. Big Data is the hottest catchphrase in current computing environment. International Data Corporation (IDC) [13] defines Big Data as: “Big Data technologies describe new generation of technologies and architectures designed to economically extract value from very large volumes of a wide variety of data, by enabling high-velocity capture, discovery and/or analysis” .

Communication systems in Big Data technology are realised by collecting and integrating data from heterogeneous services. In heterogeneous networking environment, further investigations are needed to make technology more user-centric. Research in this area includes data warehouse products with large-scale distributed data processing platforms such as Hadoop, and data analytic technologies such as machine learning and data mining.

Big Data is as an emergent technology for both practitioners and researchers, which arise significant challenges for academia, industry, and other organizations. There is a need of novel techniques to manage and analyse Big Data to create value that increase the accuracy of predictions, improve the management & security and enable informed decision making.

Earlier definitions of Big Data focused only on the structured data, but now researchers have realized that most of the information resides in semi-structured or even in unstructured format (also known as multi-structured), mainly in the form of multimedia, sensor data, XML documents, social networking data to name a few.

Vehicular Ad-hoc Network (VANET) a special form of Mobile Ad-hoc Network(MANET) is a technology that has its roots in traffic engineering and has taken enormous attention in recent years. VANET exploits vehicles as nodes in a network and involves vehicle to vehicle (V2V) and vehicle to roadside infrastructure (V2I) wireless communication [17]. Each partaking vehicle is turned into a router or wireless node that can connect and become a part of the network in the range of 100 m to 300 m approximately. Network is dropped out in case of vehicles fall out of the range. Any vehicles can join in the network, if it comes in the range to form a VANET. “Fixed infrastructure belongs to the government or private network operators or service providers” [6]. Advancing trends in ad hoc networks are envisaged to instigate a collection of wireless technologies as a type of Wi-Fi, Satellite, Cellular, Dedicated Short Range Communication (DSRC) [14] and WiMAX. VANET can be perceived as the component of the Intelligent Transportation System (ITS) [24].

Fig 1 shows the typical VANET Architecture, comprising of vehicles and road side units (RSUs) that interchange mainly safety messages. It gives the time to react for life-endangering events to the motorists. The main goal of VANET is providing safety and comfort for people and goods [20]. For this a special electronic equipment is being placed inside each vehicle to provide ad hoc network connectivity.

VANET has its roots in traffic engineering, which is concerned with traffic signal timing, traffic volume and flow counts, and other studies that determine the need for a certain roadway and safety design. In VANETs, data is collected through various vehicles and roadside units through GPS that results in increased complexity of data. By applying these Big Data technologies to Vehicular Ad-hoc Network (VANET), one can gain useful insight from a huge amount of operational data, to improve traffic management processes such as planning, engineering and operations.

Rest of the paper is organized as follows: Section II describes the characteristics of VANET and argues VANET as a Big Data problem. Big Data Analytics is discussed in section III. Routing in VANETs using Hadoop as experimental study is described in section IV and finally section V concludes the paper.

II. VANETs as a Big Data problem

With companies now turning to create digital representations of existing data and acquiring everything that is latest, data growth rates over the last few years have been very high. Many industries fall under the canopy of new data creation and digitization of existing data, and hence are becoming appropriate sources for Big Data resources. In fact, these days nearly all the sectors have started adopting digital representation of data, resulting in huge data volume. Some of the sectors which have contributed to the Big Data are Networking, Engineering, Health care, Transportation, Retail, Telecommunications, Government, Entertainment media, Video surveillance etc. Big Data can help businesses to target their potential customers and recommend services which they are craving for. Big Data has come across many issues and associated challenges like storage, scalability, processing, timeliness, privacy and security.

Big Data has been expanding very swiftly into the transportation arena i.e. VANETS. Most of the apprehensions to VANETs are of concerns in MANETs, but their minutiae varies. Somewhat marching at haphazard manner, vehicles incline to progress in a predetermined way. The associations can be depicted more accurately among roadside units. The vehicles are delimited in their scope of motion, constrained to follow a paved path.

This paper focusses on VANETs which access large data in real time for planning and management of safe and efficient traffic flow in transportation systems.

"Big Data" is characterized initially by Doug Laney [13], a META Group and now Gartner analyst as 3 V's: Volume, Variety and Velocity in 2011. In 2012 IBM finalized two more V's as Value and Veracity [10] thus making 5 V's of Big Data [26]. These 5 V's are now listed as: Volume, Variety, Velocity, Value and Veracity.

The literature survey by various authors [1], [15], [18], [19], [21], include some unique characteristics of VANETs. In this paper we have tried to map some of these VANET characteristics to attributes of Big Data to justify that VANET problems can be treated as Big Data problem and can be solved using techniques of Big Data. Various VANET characteristics we have mapped to the attributes Volume, Variety, Velocity, Value and Veracity of Big Data are as given below:

Real time data: VANET Data is real time and automatically updated after regular intervals, stored in databases resulting in large tables needed for routing decisions [15]. This maps to the *Volume* part of Big Data that refers to

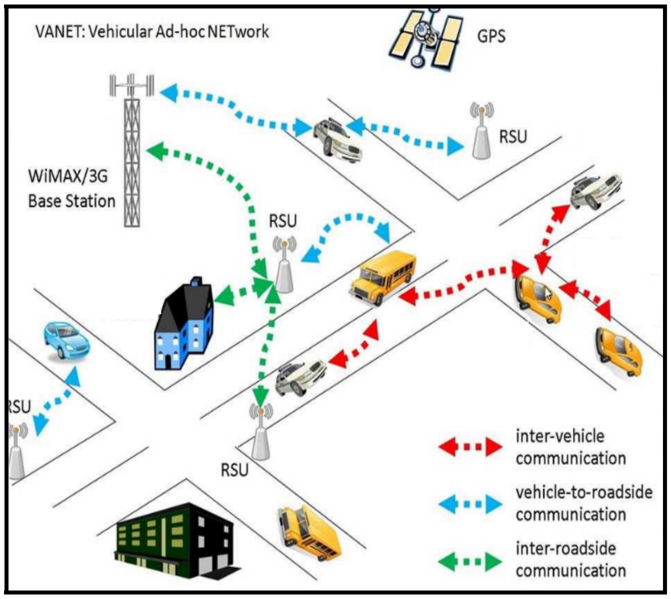

VANET: Vehicular Ad-hoc NETwork
GPS
WiMAX/3G Base Station
RSU
RSU
RSU
RSU
RSU
inter-vehicle communication
vehicle-to-roadside communication
inter-roadside communication

Fig. 1. VANET Architecture [8]

the amount of data being generated by different data sources contributing to the Big Data in real time.

- ***Variable network density:*** It depends on the highly variable vehicular density. These Vehicles are equipped with various GPS enabled sensor devices that are producing different form of data. It maps to the *Variety* part of Big Data which refers to the data coming from varied sources that are differ in the type of data being delivered by them. It includes unstructured as well as structured data.

- ***Highly Dynamic topology and Mobility modelling:*** In VANETs, the nodes refer to the vehicles that are mobile and highly dynamic in nature [21]. It results in rapid topology changes due to high speed that results in frequent fragmentation in terms of small effective diameter and network density [1], [19].This feature of VANETs justify the *Velocity* in Big Data which relates to the fact that the data is being generated at a very fast rate. It not only includes the rate of generation of data but also the time it takes to process the incoming data into the system and also the frequency with which the data is delivered.

- ***Large Scale network and high computational ability:*** Many GPS enabled devices increase the computational capacity of nodes in large scale networks [1], [18]. This require the exact value of data for predicting the routing decision. Thus, it is equivalent to *Value* part of Big Data which is important for analysing and predicting the behaviour of its nodes in order to formulate a policy or make a decision to optimize the output.

- ***Anonymous addressee and potential support from Infrastructure:*** Identification of vehicles in a particular area support from the available infrastructure is the basic requirement of the most applications in VANETs. Thus there is a requirement from the neighbouring nodes be honest in their opinion [15]. This maps to the *Veracity* part of Big Data that relates to data certainty and trustworthiness in terms of collection, processing methods, trusted infrastructure and data origin. This ensures that data used is protected from unauthorized access and modifications throughout the lifecycle.

Thus, VANETs problem can be considered as a good case study for Big Data analytics.

III. Big Data Analytics for Vanets

The research in VANETs can be used to improve the transportation system, supply chain management, and logistics by measuring thousands of variables about the roadway, vehicles, and drivers. Gathering the data requires sensors in roads and vehicles that can ascertain traffic flow, local weather conditions, how the vehicle is operating, and how the driver is responding to changing conditions. The scale here is enormous. With millions of miles of road, millions of vehicles and millions of drivers providing data collected over *years*, the sheer number of data points is astounding. It is only now, through modern Big Data technology, that this much data can be analysed in a reasonable time for it to be useful.

From this enormous choice of data, one needs to build adaptive models to help transportation companies decide on the best routes to optimize time to delivery, safety, cost, and fuel consumption. By quantifying traffic behaviour in bad weather, logistics companies can figure out alternative routes that allow delivery in time. The need is to develop predicative models of border crossing delays, a key problem for goods shipped between the various states or countries.

There is even a great need for developing models that may predict whether or not accidents are likely to occur due to road conditions and driver behaviour, eventually leading to more secure roads. The work is not the pie-in-the-sky talk about how everything will change because of some nebulous Big Data trends. This research has revealed benefits in insuring greater safety, lower transportation costs, more ecologically friendly infrastructure, and predicable material delivery for manufacturing, which is especially important for modern just-in-time manufacturing.

A single application can be generating/collecting many types of data. Data is generated fast and need to be processed fast. This data contain useful information and can be mined for getting this useful information. The traditional relational databases can't process this large volume of data. One needs new techniques to handle large volume of data. Hadoop is an open platform to process large amount of data [3].

When a company needs to store huge amount of data they can adopt two options: either procure a big machine with extra CPU, RAM, disk space etc. or consult some database vendors for a bigger solution. The two options have their own problems. There is a limit on procuring a big machine. Furthermore scaling is a challenging task. The second option infers scaling horizontally. Solution provided by this option is not cheap and requires significant investment and specialised skills. Today, "the data is being generated not only by the users and various applications and/or devices but is also machine-generated and such data is exponentially leading the change in the Big Data space. Dealing with big datasets in the order of terabytes or even petabytes is very challenging" [4]. In Modern computing environment, a widespread data processing engine used for Big Data is Hadoop, an implementation of the Map Reduce framework.

Hadoop provide reliable, shared and distributional storage through Hadoop Distributed File System (HDFS) and distributed computing capabilities through Map Reduce, a programming model.

A. HADOOP

Hadoop is an open-source framework that process very large amount of data for developing and executing various distributed applications. It specifies the methodology that distributes across a swarm of machines in a cluster and that thrusts analysis code to nodes closest to the data being analysed. Hadoop provides distributed storage called Hadoop distributed file system (HDFS) and distributed computing through a programming model called MapReduce [27], [28]. Hadoop is not an alternative for databases and data ware houses. It provides a framework for Big Data processing. The processing of structured data is much easier in relational databases.

Hadoop is one of the Big Data processing platform. This project is developed by Apache Software Foundation in Java to support distributed applications that are data intensive. Hadoop provide distributed computing and distributed storage. Hadoop enable the applications to work with millions of nodes and yottabytes of data. Google File System and Google's MapReduce papers inspire to work with Hadoop. Hadoop is used widely at many business platforms like search Yahoo as its biggest contributors. The two main components of Hadoop are HDFS and MapReduce.

B. Hadoop Distributed File System (HDFS)

A HDFS cluster is working in master-worker pattern with two types of node: a NameNode (the master) and a number of DataNodes (workers). File system namespace is managed by the NameNode. It also asserts the file system tree and the metadata of all the files and directories in the tree. The workhorses of the file system are DataNodes. They collect and recover blocks when they are conveyed to (by clients or the NameNode), and they convey back to the NameNode periodically with lists of blocks that they are collecting.

The replication of data blocks is decided by NameNode. The default block size is 64MB in a typical HDFS with a replication factor of 3 (local rack holds second copy and remote rack holds third copy).The replication factor can be changed according to the user wish. HDFS have default replication policy such that each DataNode contains at most one replica of any block and each rack contain at most two replicas of the same block, as long as there are sufficient racks on the cluster. Each block has sufficient number of replicas is ensured by the NameNode. It will continuously check whether the block is over-replicated or under-replicated. If it is over-replicated, NameNode will select a replica to remove. If it is under-replicated, NameNode will create another replica in any node or in the same node. The replica removal or replica creation is done based on the replication policy.

To read a file from the HDFS, the client first contacts the NameNode and gets the information such as where are the blocks of the needed file and replicas of each node residing. Then the client reads the block from a node which is closer to the client. In order to write a file again the client needs to contact the NameNode and gets the permission for a write. Then the client writes the data into any one of the node and passes the block content into another node or same node in order to provide replication.

During each operation, the DataNode sends a heartbeat message to the NameNode indicating that the DataNode is still alive and the block replicas are available. If the NameNode doesn't get any message from a DataNode within a particular time, then that DataNode is considered to be out of service and the replicas residing on that DataNode are copied into another node.

C. *Map Reduce*

MapReduce is a linearly scalable programming model introduced by Google that makes it easy to process in parallel massively large data on large number of computers. MapReduce works mainly through two functions, Map function and Reduce function. Map function takes a set of key/value input and map it into zero or more set of key/value. The reduce function then takes each unique key and group its associated values into a unique key/value set [9]. Map Reduce provides ease-of-use, scalability, and failover properties.

Map function: The map function is summarized as follows [9],

"*Map, written by the user, takes an input pair and produces a set of intermediate key/value pairs. The MapReduce library groups together all intermediate values associated with the same intermediate key (I) and passes them to the reduce function.*"

We can further say,

$$\text{Map (k1, v1)} \longrightarrow \text{list (k2, v2)}$$

The input keys and values have different domains as compared to the domains to which the intermediate keys and values belong.

Reduce function: The reduce function can be summarized as follows [16],

"*The reduce function, also written by the user, accepts an intermediate key (I) and a set of values for that key. It merges together these values to form a possibly smaller set of values. Typically just zero or one output value is produced per reduce invocation. The intermediate values are supplied to the user's reduce function via an iterator. This allows us to handle lists of values that are too large to fit in memory.*"

We can further say,

$$\text{Reduce (k2, list (v2))} \longrightarrow \text{(k2, v3)}$$

The intermediate keys and values have the same domain as the output keys and values. But, the original input keys and values have different keys and values as compared to the final output keys and values.

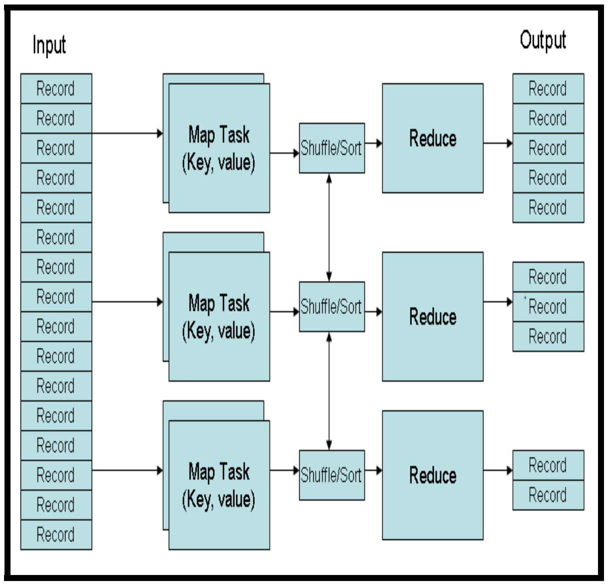


Fig. 2. Map Reduce Architecture

D. *Execution overview of Hadoop*

The Map Reduce library first splits the input files of User program into M pieces of normally 16MB to 64MB per piece. Then, many copies of the program on a cluster of machines are initiated. All the functions are controlled by the master. The rest of the nodes that are assigned work by the master are workers. There are M map tasks and R reduce tasks to assign. The idle workers are picked by the master and assigns each one a map task or a reduce task. The contents of the corresponding input file are read by a worker that is assigned a map task. It parses key/value pairs out of the input data and passes each pair to the user-defined map function. The map function generates the intermediate key/value pairs and buffered them in memory. Sporadically, these buffered pairs are noted to local disk. The partitioning function then partitioned them into R regions. The output of the Map is stored in the local disk temporally which is removed as soon as the Reduce function is completed. This temporary storage of Map output helps in fault tolerance and recovery. That is why, when a Reduce function fails, the system need not start the Map process again. The Map output is then passed back to the master on the local disk. The master is liable for redirecting these locations to the reduce workers.

After getting a notification about these locations from the master, the reduce worker read the buffered data from the local disks of the map workers through remote procedure calls. Reduce worker reads the intermediate data for its partition and sort it by the intermediate keys so all occurrences of the same key are grouped together. The sorting is needed because typically many different keys map to the same reduce task. If the amount of intermediate data is too large to fit in memory, an external sort is needed. The reduce worker iterates over the sorted intermediate data. Then the user's reduce function is given the key and corresponding set of intermediate values. In case of a system failure, completed reduce tasks do not need to be re-executed since their output is stored in a global system. For this reduce partition, output of the reduce function is added to a final output file. The master stimulates the user program after the completion of all map and reduce tasks,. The typical Map Reduce framework is shown in Fig 2.

IV. Routing In VANETs Using Hadoop

The goal of VANET is to establish an efficient route between nodes i.e. vehicles and also to adjust this efficiently for varying topology of moving vehicles. Thus, Routing has always been a major challenge in VANETs due to mobility and frequent changes in the network topology. To achieve this a routing protocol is needed that can direct the way through which the information can be exchanged between two communication entities and incorporates the procedure of a route establishment, forwarding decision information and action in maintaining the route and/or recovering from the routing failure.

"Dijkstra's algorithm, was perceived by the Dutch computer scientist Edsger Dijkstra in 1956 and published in 1959" [11], [7]. "For a given source node in the graph, the algorithm finds the path with lowest cost (i.e. the shortest path) between that node and every other node. For example, if the nodes of the graph represent cities and edge path costs represent driving distances between pairs of cities connected by a direct road, Dijkstra's algorithm can be used to find the shortest route between one city and any other city"[11],[7].

A major assumption for the graph search algorithm is to have a single-source that is solving shortest path problem for a graph with nonnegative edge path costs and producing a shortest path tree. Other assumptions include the availability of sufficient number of routes, communication links, presence of two way radio, infrastructure such as GPS, existence of Road side units (RSU) at appropriate positions for propagation of signal [5], [23], [25].

The underlying assumption is that VANET network topology can be viewed as a directed graph with vehicles representing nodes and shortest path representing links in the graph. The Dijkstra's shortest path algorithm is a type of Geographic Source Routing (GSR) algorithm based on position for VANETs [12]. "GSR assumes the aid of a street map in city environments to know the city topology. GSR combines geographic routing and topological knowledge from street maps. The sender determines the junctions that have to be traversed by the packet using the Dijkstra's shortest path algorithm and then forward the packet in a position-based fashion between the junctions" [2], [22]. This algorithm can be efficiently coded using Big Data analytics.

In this paper, the performance of Dijkstra algorithm in Vehicular Ad-hoc Networks is evaluated on Big Data framework i.e. Hadoop platform open environment running in standalone node as well as on multi-node cluster with 2, 3, 4 and 5 nodes respectively. The results are also compared using NetBeans IDE 7.4 environment running on a Windows platform. The results in terms of processing time are tabulated by increasing the number of nodes in the system. The algorithm was run for a graph consisting of 100 nodes initially. Then we increased the number of nodes with a step of 100, i.e. 200, 300, 400 and 500 in each subsequent iteration.

V. Implementation And Results

The Dijkstra's algorithm was implemented in JAVA for single standalone distributed Hadoop map-reduce framework and evaluated for the given number of nodes then further evaluated on multinode fully distributed Hadoop cluster environment with 2, 3, 4 and 5 nodes respectively.

In Map Reduce programming, the task is performed by dividing the entire program into smaller program and executing them independently. Many data analysis algorithms are involved in parallelization of MapReduce method for ease-of-use and effectiveness. "An important characteristic of Hadoop is the partitioning of data and computation across thousands of nodes, and the execution of application computations in parallel close to their data. A Hadoop cluster scales computation capacity, storage capacity and I/O bandwidth by simply adding commodity servers" [16]. It can use the existing commodity of hardware and therefore the cost of setting Hadoop cluster is relatively low.

For running a Map Reduce program a small Hadoop cluster is needed that may include a single master node and multiple worker nodes. "The master node consists of a JobTracker, TaskTracker, NameNode and DataNode. A slave or worker node acts as both a DataNode and TaskTracker, though it is possible to have data-only worker nodes and compute-only worker nodes" [16].

The flow of steps involved in MapReduce programming model are as follows [16]:

i. Iteration over the input

ii. Computation of key/value pairs from each piece of input

iii. Grouping of all intermediate values by key

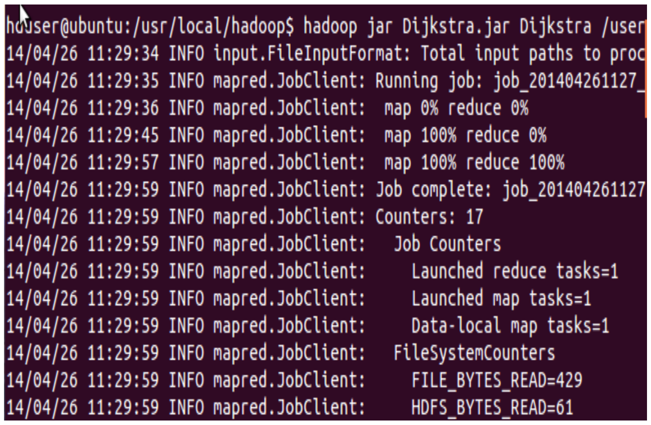
iv. Iteration over the resulting groups

v. Reduction of each group

The computation receives a set of input key/value pairs, and yields a set of output key/value pairs. Two functions, map and reduce, are used to express the computation. In this implementation, we took starting node as the *key* and distance it points to as the *value* in Mapper part and current node as the *key* and minimum of all the possible distances to this point as the *value* in Reducer part for key/value pair.

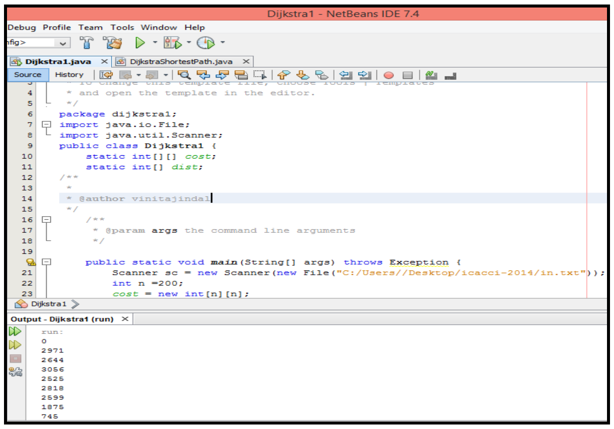
Results are tabulated in Table 1 which shows the time taken (in Seconds) for a job in different cases with increased number of nodes. It was observed that in NetBeans IDE 7.4 with 100 nodes, the job takes 1.1 seconds to complete, while in Hadoop standalone node it takes 1.2 seconds to complete the same task. Further Hadoop distributed system with 2, 3, 4, 5 nodes it takes 1.2 seconds, 1.1 second, 1.0 second and 0.8 seconds respectively to execute the job. Both output screen shots are displayed in Fig 3 and Fig 4 respectively and final results are shown graphically in Fig 5.

It is inferred from the results that when the number of nodes in graph increases, the processing time of the job in Hadoop environment tends to decrease with the increased number of cluster nodes due to distributed processing of the data on different nodes and the difference in processing time increases with existing Java scenario as the number of nodes in Hadoop environment increase.



```
hduser@ubuntu:/usr/local/hadoop$ hadoop jar Dijkstra.jar Dijkstra /user
14/04/26 11:29:34 INFO input.FileInputFormat: Total input paths to proc
14/04/26 11:29:35 INFO mapred.JobClient: Running job: job_201404261127_
14/04/26 11:29:36 INFO mapred.JobClient:  map 0% reduce 0%
14/04/26 11:29:45 INFO mapred.JobClient:  map 100% reduce 0%
14/04/26 11:29:57 INFO mapred.JobClient:  map 100% reduce 100%
14/04/26 11:29:59 INFO mapred.JobClient: Job complete: job_201404261127
14/04/26 11:29:59 INFO mapred.JobClient: Counters: 17
14/04/26 11:29:59 INFO mapred.JobClient:   Job Counters
14/04/26 11:29:59 INFO mapred.JobClient:     Launched reduce tasks=1
14/04/26 11:29:59 INFO mapred.JobClient:     Launched map tasks=1
14/04/26 11:29:59 INFO mapred.JobClient:     Data-local map tasks=1
14/04/26 11:29:59 INFO mapred.JobClient:   FileSystemCounters
14/04/26 11:29:59 INFO mapred.JobClient:     FILE_BYTES_READ=429
14/04/26 11:29:59 INFO mapred.JobClient:     HDFS_BYTES_READ=61
```

Fig. 3. Snapshot of output on hadoop map-reduce framework.


Dijkstra1 - NetBeans IDE 7.4
Debug Profile Team Tools Window Help
Dijkstra1.java
DijkstraShortestPath.java
Source
History
Dijkstra1
Output - Dijkstra1 (run)

```
 * and open the template in the editor.
 */
package dijkstra1;
import java.io.File;
import java.util.Scanner;
public class Dijkstra1 {
    static int[][] cost;
    static int[] dist;
/**
 *
 * @author vinitajindal
 */
    /**
     * @param args the command line arguments
     */

    public static void main(String[] args) throws Exception {
        Scanner sc = new Scanner(new File("C:/Users//Desktop/icacci-2014/in.txt"));
        int n =200;
        cost = new int[n][n];
```

```
run:
0
2971
2644
3056
2525
2818
2599
1875
745
```

Fig. 4. Snapshot of output in NetBeans IDE 7.4

VI. CONCLUSION

Big Data is catching the attention of researchers in almost all areas as a lot of data is being generated by various day-to-day applications these days. Volume, Variety, Velocity, Veracity and Value are the attributes associated with Big Data in literature. In this paper, the problem of VANET is discussed and the mapping of some VANET characteristics to Big Data attributes is done to substantiate that VANET problem can be considered as a Big Data problem and various techniques of Big Data can be applied to VANET data in order to improve traffic management processes In this paper, Dijkstra's algorithm used in Vehicular Ad-hoc Networks for routing is taken as a case study to be solved using Big Data technologies. The performance of Dijkstra algorithm on Big Data framework using standalone distributed Hadoop Map Reduce environment as well as Hadoop distributed environment with 2, 3, 4 and 5 nodes is also evaluated. The results obtained from the experiments were compared with results of conventional NetBeans IDE environment with respect to increase in the number of nodes in the system. It was found that increasing the number of cluster nodes using Hadoop framework, processing time for the routing algorithm is greatly reduced.

TABLE I. TIME TAKEN (IN SECONDS) BY DIJKSTRA'S ALGORITHM

Number of nodes in the system	**Time taken by Net Bean IDE 7.4**	**Time taken by map Reduce Hadoop standalone distributed system**	**Time taken by map Reduce Hadoop distributed system with 2 nodes**	**Time taken by map Reduce Hadoop distributed system with 3 nodes**	**Time taken by map Reduce Hadoop distributed system with 4 nodes**	**Time taken by map Reduce Hadoop distributed system With 5 nodes**
100	1.1	1.2	1.2	1.1	1	0.8
200	2.4	1.8	1.7	1.6	1.4	1.1
300	3.2	2.3	2.1	1.8	1.7	1.5
400	4.6	2.9	2.6	2.3	2.1	1.9
500	5.3	3.1	2.9	2.6	2.5	2.2

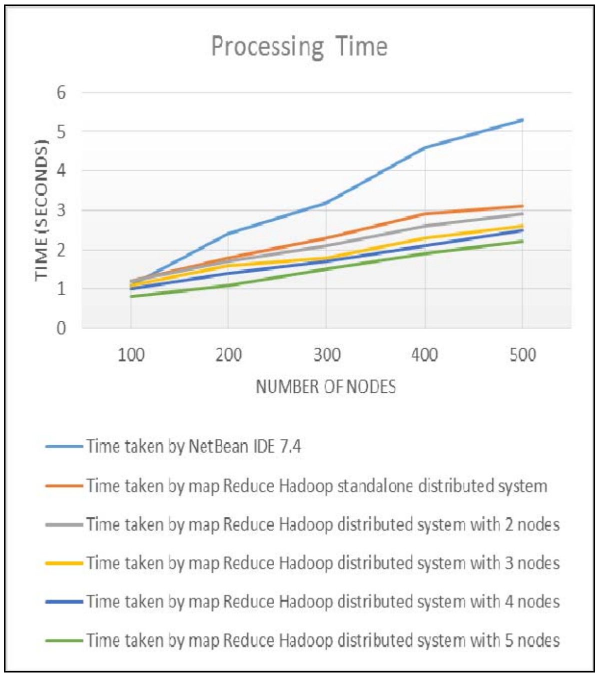


Fig. 5. Processing Time verses number of nodes in system

ACKNOWLEDGMENT

The authors duly acknowledge the University of Delhi for the support in work on this paper under the research grant number DRCH/R&D/2013-14/4155.

REFERENCES

[1] Al-Sultan, S., Al-Doori, M. M., Al-Bayatti, A. H., & Zedan, H. (2014). "A comprehensive survey on vehicular Ad Hoc network. Journal of Network and Computer Applications, 37, 380-392.

[2] B, S. S., G, N. K., Rani, U. H., K, D. C., George, G., & Murali, A. (2012). GPS Based Shortest Path for Ambulances using VANETs. 2012 International Conference on Wireless Networks (ICWN 2012) (pp. 190-196). Singapore: IPCSIT vol. 49, IACSIT Press.

[3] Barlow, M. (2013). Real-Time Big Data Analytics: Emerging Architecture. Sebastopol: O'Reilly.

[4] Barské, A. (2013). Big Data Whitepaper,. Erdogan: Arzu Barské - Erdogan 2013 © version 2.0.

[5] Cho , K. H., & Ryu, M. W. (2012). A Survey of Greedy Routing Protocols for Vehicular Ad Hoc Networks. Smart Computing Review, vol. 2, no., 125-137.

[6] Consortium, C.-t.-C. C. (2014, April 4). Retrieved from http:/www.car-to-car.org.

[7] Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2001). Introduction to Algorithms. MIT Press and McGraw-Hill.

[8] Corser, G. (2013, May 9). Retrieved from https://www.youtube.com/watch?v=DrH-1505-Mg

[9] Dean, J., & Ghemawat, S. (2008). MapReduce : Simplified Data Processing on Large Clusters. Communications of the ACM, 51(1), 1-13.

[10] Demchenko, Y., Worring, M., Los, W., & Laat, C. d. (2013, september 16). 2013-09-16-rda-bdaf.pdf. Retrieved December 11, 2013, from http://www.delaat.net: http://www.delaat.net/~cees/posters/2013-09-16-rda-bdaf.pdf

[11] Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. In 1. Numerishe Mathematik.

[12] Gabriel, J., & Lopez, B. (2010). A survey of geographic routing protocols for Vehicular Ad Hoc Networks (VANETs). Academia.com.

[13] Gantz, J., & Reinsel, D. (2011). The 2011 Digital Universe Study: Extracting Value from Chaos.

[14] Home, D. S. (2014, February 25). Retrieved from http://www.leearmstrong.com/DSRC/DSRCHomeset.htm

[15] Karim, R. (2008). VANET: Superior System for Content Distribution in Vehicular Network Applications. Rutgers University, Department of Computer Science, Tech. Rep.

[16] Lämmel, R. (2008). Google's MapReduce programming model—Revisited. Science of computer programming, 70(1), 1-30.

[17] Li, Z., Wang, Z., & Chigan, C. (2011). Security of Vehicular Adhoc Networks. In Wireless Technologies in Intelligent Transportation Systems (pp. 133-174). Nova Publishers.

[18] Mohapatra, P., & Krishnamurthy, S. V. (2005). AD HOC NETWORKS : Technologies and Protocols. Boston, USA: Springer.

[19] Offor, P. I. (2012). Vehicle Ad Hoc Network (VANET): Safety Benefits and Security Challenges . Social Science Research Network. doi:http://dx.doi.org/10.2139/ssrn.2206077

[20] Paul, B., Ibrahim, M., & Bikas, M. N. (2011). VANET Routing Protocols: Pros and Cons. International Journal of Computer Applications.

[21] Qian , Y., & Moayeri, N. (2008). Design of Secure and Application-Oriented VANETs. Vehicular Technology Conference, 2008. VTC Spring 2008 (pp. 2794 - 2799). Singapore: IEEE.

[22] Ryu, M. W., & Cho, K. H. (2012). A Survey of Greedy Routing Protocols for Vehicular Ad Hoc Networks. Smart Computing Review, vol. 2, no. 2, 125-137.

[23] Saha, A. K., & Johnson, D. B. (2004). Modelling Mobility for Vehicular Ad hoc Networks. VANET'04. USA: ACM.

[24] Shivani, & Singh, J. (2013). Route Planning in Vanet By Comparitive Study of Algorithms. International Journal of Advanced Research in Computer Science and Software Engineering, Volume 3, Issue 7, 682-689.

[25] Transporatation, U. D. (2014, january 25). Intelligent Transportation System (ITS) Home. Retrieved from http://www.its.dot.gov/index.htm

[26] Wang, W., Xie, F., & Chatterjee, M. (2009). Small-Scale and Large-Scale Routing in Vehicular Ad Hoc Networks. IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, VOL. 58, NO. 9,, 5200-5213.

[27] Warden, P. (2012). Big Data Glossary. Sebastopol: O'Reilly Media, Inc.

[28] Zikopoulos, P. C., Eaton, C., deRoos, D., Deutsch, T., & Lapis, G. (2012). Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. United States of America : McGraw-Hill.