Public Transportation BigData Clustering

Preliminary Communication

Tomislav Galba
J.J. Strossmayer University of Osijek
Faculty of Electrical Engineering
Cara Hadriana 10b, 31000 Osijek, Croatia
tomislav.galba@etfos.hr

Zoran Balkić
J.J. Strossmayer University of Osijek
Faculty of Electrical Engineering
Cara Hadriana 10b, 31000 Osijek, Croatia
zoran.balkic@etfos.hr

Goran Martinović
J.J. Strossmayer University of Osijek
Faculty of Electrical Engineering
Department of Electromechanical Engineering
Cara Hadriana 10b, 31000 Osijek, Croatia
goran.martinovic@etfos.hr

Abstract *–An increase in the use of GPS modules and cell phones with location services has created a need for new ways of collecting and storing data. Considering a fairly large number of devices, data collected in such way in most cases take up a vast amount of space on servers while on the other hand, they represent a source of very useful information. A large number of companies use this method of data collection in order to create prediction models, reports and data analysis. As an object of observation, we use the database of a modern public transportation system which contains information about vehicle telemetry. In this paper, we will describe the application and result analysis of some well-known clustering algorithms in order to solve public transportation problems like traffic congestion, passenger transport, etc.*

Keywords *– analysis, big data, clustering, GPS, public transportation*

1. INTRODUCTION

Widespread use of GPS modules and cell phones with location services allows us to store a very large amount of useful spatiotemporal data that can later be used for analysis and creation of prediction models. The amount of data stored in the world database increases at a very high rate on an annual basis [1]. Many governments and organizations perform tests on BigData because of their importance in several application domains such as financial, communication, public transportation, public health and other services. Since our object under consideration is a modern public transportation system, collected data consists of vehicle telemetry.

In this paper, we focus on discovery of traffic congestion based on collected public transportation vehicle telemetry data [2]-[4]. Congestion can be caused by poorly timed traffic lights, traffic accidents or bad bus route organization. Clustering of data will be performed by using two well-known clustering algorithms, i.e., the K-means and the DBSCAN clustering algorithm. The organization of a large number of data points into classes or clusters based on their similarity is called cluster analysis [5].

The rest of the paper is organized as follows. Section 2 describes each of the listed clustering algorithms. In Section 3, we describe a data structure, the method for data manipulation and results. Section 4 concludes the paper.

2. CLUSTERING ALGORITHMS

Data clustering is the unsupervised classification of patterns into groups where N objects within same group are more similar to their neighbors than those in other groups [6]. It is very useful in data mining, image segmentation, pattern classification, etc.

Clustering can be achieved by various algorithms that are categorized based on their cluster model. Typical cluster models are:

- Connectivity model,
- Centroid model,
- Distribution model,
- Density model,
- Subspace model,
- Group model,
- Graph-based model.

For the purpose of analysis, in this paper we use two different data clustering algorithms, i.e., the K-means algorithm based on the centroid model, and DBSCAN (Density-Based Spatial Clustering of Applications with Noise) based on the density model.

2.1. K-MEANS CLUSTERING

The K-means algorithm is a well-known and one of the simplest unsupervised clustering method used. It is used to classify a given dataset through a fixed number of clusters (*k* clusters). Hence, naturally, as an input parameter it takes a *k* value which represents the number of k clusters, and then partitions a set of *n* objects into *k* clusters.

The resulting clusters are compared for similarity according to the mean value of the object in the cluster. The algorithm proceeds as follows:

- For a given set of data points $X=\{\{x_1,y_1\},\{x_2,y_2\},\ldots\{x_n,y_n\}\}$, we randomly select cluster centers. Since different cluster centers give different results, it is recommended to place them as far away from each other as possible.
- Calculate the distance matrix between the assigned cluster centers and each data point. Based on the distance matrix data points are assigned to the cluster with the closest centroid.
- Recalculate centroid values. If there is more than one data point assigned to a certain centroid, its value is updated with an average value of assigned data points.
- Repeat Steps 2 and 3 until there is no difference between cluster groups or the difference is within 1%.

Fig.1 shows how initial centroids in the K-means algorithm move as recalculations are conducted.

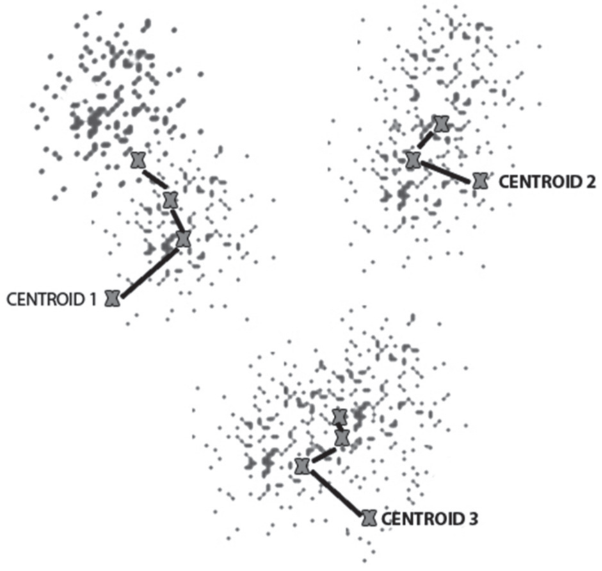

Fig. 1. K-means clustering

2.2. DBSCAN CLUSTERING

DBSCAN is a data clustering algorithm proposed in [7]. It is a density-based algorithm because it can identify clusters in a large spatial dataset by looking at the local density of corresponding elements. The advantage of the DBSCAN algorithm over the K-means algorithm which is mentioned in this paper is that the DBSCAN algorithm can determine which data points are noise or outliers. Fig. 2 shows the ability of the DBSCAN algorithm to detect the shape of a cluster in contrast to the k-means algorithm that we also used in this paper.

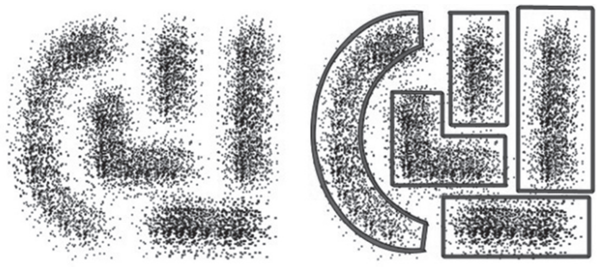

Fig. 2. DBSCAN cluster detection

The DBSCAN algorithm computing process is based on the following six definitions [8], [9]:

1.The Eps neighborhood of a point

$$N_{Eps}(p) = \{q \in D | dist(p,q) \leq E_{ps}\} \quad (1)$$

2. Directly density-reachable

$$p \in N_{Eps}(q) - \mathit{border\ point\ condition} \quad (2)$$

Point *p* is declared as a border point if the number of neighboring points is less than MinPts but within limits of the core point radius.

$$|N_{Eps}(q)| \geq MinPts - \mathit{core\ point\ condition} \quad (3)$$

Point *p* is declared as a core point if the number of neighboring points is greater than or equal to MinPts.

3. Density-reachable - point *p* is density-reachable from point q if the array of points p is such that *array[i+1]* is directly density-reachable from *array[i]*.

4. Density-connected – points *p* and *q* are density-connected if there is some point *x* which is density-reachable from *p* and *q*.

5. Cluster - if point *p* is part of cluster *C* and point *q* is density-reachable from point *p*, then *q* is also part of cluster *C*.

6. Noise – a set of points that are not assigned to any of the clusters.

Fig. 3 shows graphical representation of points based on the aforementioned 6 definitions.

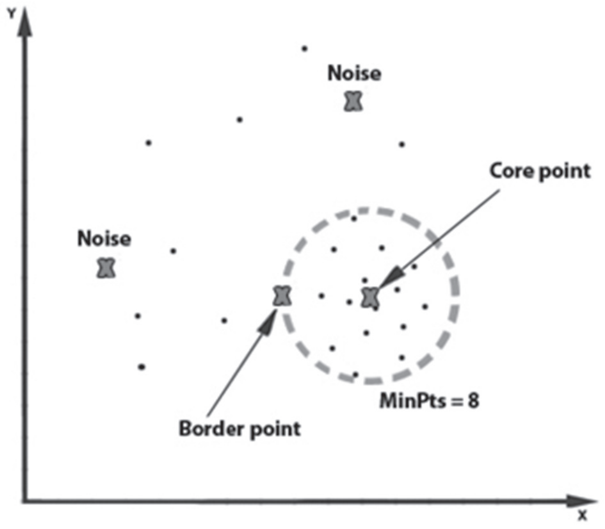

Fig. 3. DBSCAN clustering definitions

The algorithm proceeds as follows:

- Definition of MinPts and radius variables.
- Calculation of the distance matrix for all data points.
- If the number of neighboring points is greater than or equal to MinPts variable, the core point is added (as shown in (3)).
- If the number of neighbors is less than MinPts variable and the distance from the core point is less than the radius, then the boundary point is added.
- If the number of neighbors is less than MinPts and not a boundary, then the outlier point is added.
- Repeat Steps 3 to 5 until all of the points have been processed.

2.2. DISTANCE CALCULATION

Since both data clustering algorithms listed above use the distance matrix for a given dataset, we use two different formulas for distance calculation [10] between two points.

Euclidean distance between two data points equals the square root of the sum of the squares of the differences between corresponding values. Therefore, if *(p, q)* are two points in the Cartesian coordinate system, the distance d in two-dimensional space from *p* to *q* and vice versa is:

$$d\,(q,p) = \sqrt{(q_1-p_1)^2+(q_2-p_2)^2+\cdots+(q_n-p_n)^2} = \sqrt{\sum_{i=1}^{n}(q_i-p_i)^2} \qquad (4)$$

and for three-dimensional space:

$$d\,(q,p) = \sqrt{(q_1-p_1)^2+(q_2-p_2)^2+(q_3-p_3)^2} \qquad (5)$$

Haversine distance is an equation which gives the great-circle distance between two data points from their longitude and latitude values considering spherical properties of the earth. Because earth is not a perfect sphere, radius *R* varies from 6356.78 km at the poles to 6378.14 km at the equator, so the mean value of *R* = 6367.45 km is used.

$$d(p,q) = 2r\arcsin\left(\sqrt{sin^2\left(\frac{\Phi_2-\Phi_1}{2}\right)+\cos(\Phi_1)\cos(\Phi_2)\,sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right), \qquad (6)$$

where,

$$r - radius\ of\ the\ sphere$$

$$\Phi_1, \Phi_2 - \ latitude\ of\ p1\ and\ p2$$

$$\lambda_1, \lambda_2 - longitude\ of\ p1\ and\ p2$$

3. EXPERIMENT AND RESULTS

For our clustering task we used one dataset which contains records of vehicle telemetry in the City of Osijek, Croatia. The dataset was provided by an urban passenger transportation company in Osijek. Data consists of positions of the vehicles retrieved from the GPS module in the period from 25 July 2008 to 25 August 2012, all stored into traditional RDBMS. The data were recorded while the vehicles were both in motion and at a stand-still position. Each record contains a vehicle ID, date and time, latitude and longitude values, speed, moving angle and signal information shown in Table 1.

Given that this is big data, it is desirable to divide data into smaller chunks. Besides that, raw data does not contain accurate information like false or unnecessary reading that must be processed in order to get relevant results.

Since we observe traffic congestion in the City of Osijek, first we need to filter and remove all data with move speed greater than 0 in order to get the result of all vehicles at a stand-still position.

Table 1. Example of raw data from the GPS module

ID	Time	Long/Lat	Movement speed	Angle	Signal
1000008	2008-07-25 12:41:10	18.7019100 ,45.5600281	0	198	1
1000002	2008-07-25 12:30:49	18.6614819 ,45.5599518	0	124	1
1000003	2008-07-25 12:41:00	18.6339207 ,45.5651169	0	62	1
1000004	2008-07-25 12:40:58	18.6311169 ,45.5339508	38	161	1

After we removed all points where speed is greater than 0, the remaining points represent all buses standing on bus stops , as shown in Fig.4, or standing still in some kind of traffic jam like traffic lights or other. In order to get relevant results we need to filter all points which have to do only with standing still at traffic lights or in any other type of traffic jam.

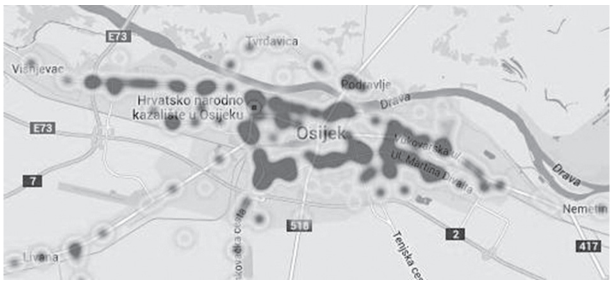

Fig. 4. Current stops heatmap for the City of Osijek

As a rule, we determined that each point must be located minimum 20 meters from the stop, as shown in Fig. 5. In this way, we avoided situations where the stop is right next to the traffic lights. Results of filtered data points are listed in Table 2.

Both clustering algorithms mentioned in this paper were implemented by taking the prepared dataset shown in Table 2 and using the Java programming language.

Table 2. Number of data points after filtering

Number of raw data points	Movement speed equals 0	Movement speed equals 0 with excluded stops
69,999,497	6,007,512	3,983,456

Fig. 6 shows initial data points for each vehicle (X axis) and their number (Y axis). This data needs further processing in terms of removing the same data points for each vehicle. Better results can be achieved in this way in terms of speed with no impact on the quality of results. Fig. 7 shows filtered data points with the same value where the total number of data points is reduced.

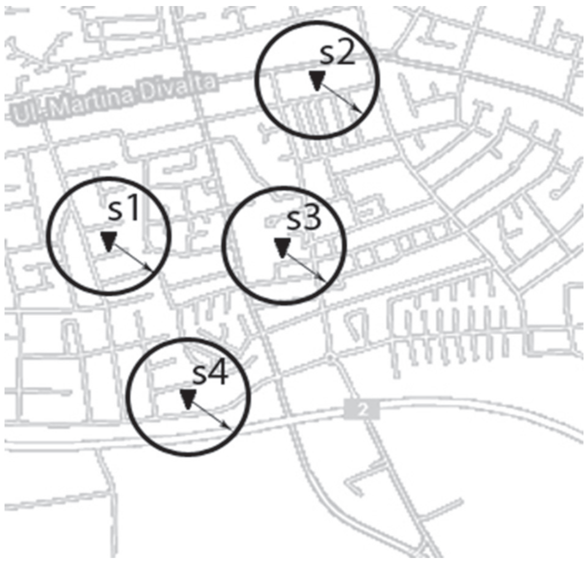

Fig. 5. Exclusion of points near stops

The overall results of data clustering are shown in Fig. 8. We can clearly see focal points where traffic congestion is most serious. The DBSCAN algorithm has performed clustering by taking the distance threshold 500 meters and the minimum number of neighbors as 2500. 8.03% of data points was classified as noise. The K-means algorithm performed clustering with 12 predefined centroids and an average of 50 iterations.

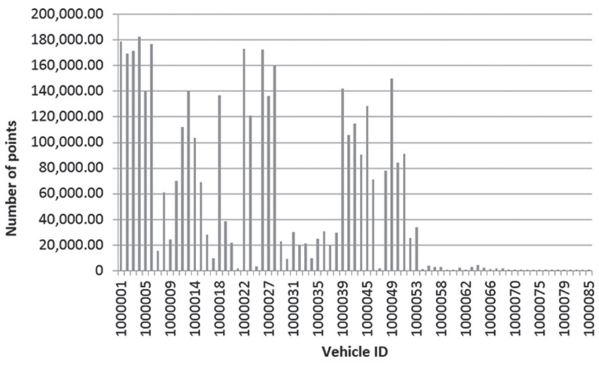

Fig. 6. Number of data points for each vehicle ID

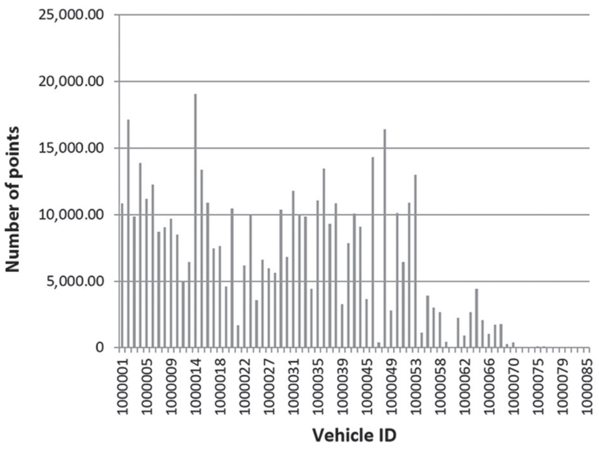

Fig. 7. Number of data points for each vehicle ID after additional filtering

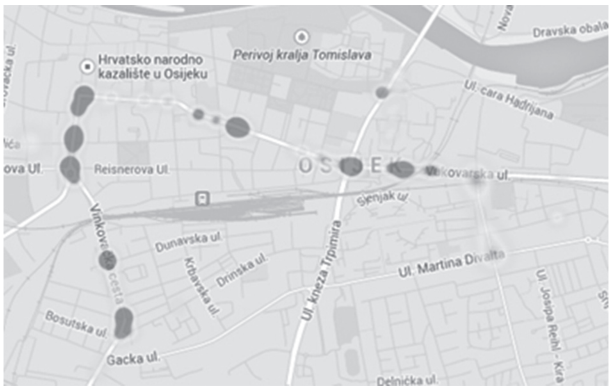

Fig. 8. Congestion heatmap

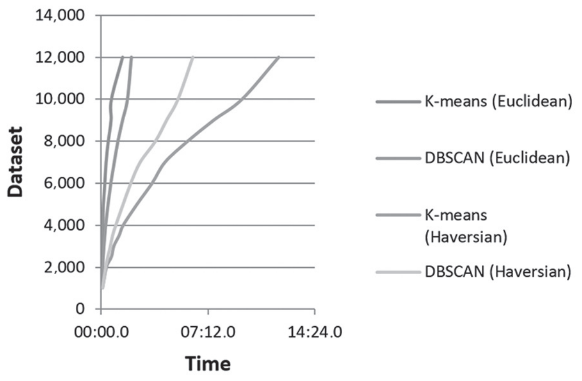

Fig. 9. The performance of clustering algorithms

Since we used more than one formula for distance calculations, we also conducted few tests related to the performance of each clustering algorithm depending on the distance calculation formula used. Fig. 9 shows the overall results of performance testing based on the maximum of 12,000 data points. The test was conducted for both clustering algorithms using both distance calculation formulas mentioned in this paper. From the results we can conclude that the use of the Euclidean distance formula will give better time in both the K-means and the DBSCAN algorithm. Still, if there are points with a greater distance, the use of a certain distance calculation formula needs to be considered due to known disadvantages. For example, Euclidean distance calculation can be conducted on data points that are relatively close to each other. On the other hand, Haversine distance calculation takes the spherical property of the earth into consideration so the distance between data points have less impact on the results in comparison with the Euclidean formula.

4. CONCLUSION

Cluster analysis divides first an unrelated set of data points into meaningful groups (clusters) of data. But, in some cases data clustering can be used as an input for further processing. Due to its practical problem application cluster analysis has long been used in a variety of fields such as biology, pattern recognition, image segmentation, machine learning, data mining, etc. In this paper, we have presented a study of congestion in the City of Osijek based on real data in the period of four years using the aforementioned clustering algorithms. This type of clustering can also be used to identify other points of interest in public transportation such as fuel consumption, passenger flow, traffic prediction, etc. It is also a very good solution to creating prediction models based on a very large collection of previously collected data.

5. REFERENCES

[1] Z. Zamani, M. Pourmand, M. H. Saraee , Application of Data Mining in Traffic Management: Case of City of Isfahan, 2010 Int. Conf. on Electronic Computer Technology, Kuala Lumpur, Malaysia, 7-10 May 2010, pp. 102-106.

[2] T. Holleczek, L. Yu, J.K. Lee, O. Senn, K. Kloeckl, C. Ratti, P. Jaillet, Detecting Weak Public Transport Connections from Cellphone and Public Transport Data, ASE Int. Conf. on Big Data Science and Computing, 4-7 August 2014, Beijing, China, pp. 1-8.

[3] T. Diamantopoulos, D. Kehagias, F.G. König, D. Tzovaras, Use of Density-based Cluster Analysis and Classification Techniques for Traffic Congestion Prediction and Visualisation, Transport Research Arena (TRA), http://www.ecompass-project.eu/?q=node/316 (accessed: 15 May 2013)

[4] B. Agard, C. Morency , M. Trépanier, Mining Public Transport User Behaviour from Smart Card Data, 12th IFAC Symposium on Information Control Problems in Manufacturing, Saint Etienne, France, 17-19 May 2007, pp. 399-404.

[5] A.K. Akasapu, P.S. Rao, L.K. Sharma, S.K. Satpathy, Density Based k-Nearest Neighbors Clustering Algorithm for Trajectory Data, Int. Journal of Advanced Science and Technology, Vol. 31, No. 1, 2011 , pp. 47-57.

[6] Dr. Ch. G.V.N. Prasad, K.H. Rao, D. Pratima , B.N. Alekhya, Unsupervised Learning Algorithms to Identify the Dense Cluster in Large Datasets, International Journal of Computer Science and Telecommunications, Vol. 2, No. 4, 2011, pp. 26-31.

[7] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise, 2nd Int. Conf. on Knowledge Discovery and Data Mining, August 2–4, 1996, Portland, OR, USA, pp. 226-231.

[8] H. Bäcklund, A. Hedblom, N. Neijman, A Density-Based Spatial Clustering of Application with Noise,http://staffwww.itn.liu.se/aidvi/courses/06/dm/Seminars2011, Linköpings Universitet – ITN, Sweden (accessed: 10 May 2013).

[9] D. Birant, A. Kut, ST-DBSCAN: An Algorithm for Clustering Spatial-Temporal Data, Data & Knowledge Engineering, Vol. 60, No. 1, 2007, pp. 208-221.

[10] F. Ivis, Calculating Geographic Distance: Concepts and Methods, Canadian Institute for Health Information, Toronto, Ontario, Canada, http://www.lexjansen.com/cgi-bin/xsl_transform.php?x=sda&c=nesug (accessed: 27 April 2013)