International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 6, No.1 (Jan-2017)
http://dx.doi.org/10.12785/ijcds/060104

A Study of Pre-Processing Technique for Map-Matching Schemes of GPS-Enabled Vehicles

Aftab Ahmed Chandio[1, 2], Fan Zhang[2], Zubair Ahmed Kalhoro[1], Imtiaz Ali Korejo[1] and Muhammad Saleem Chandio[1]

[1] *Institute of Mathematics & Computer Science, University of Sindh, Jamshoro 70680, Pakistan*
[2] *Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen 518055, China*

Received 23 Sep. 2016, Revised 15 Nov. 2016, Accepted 3 Dec. 2016, Published 1 Jan. 2017

Abstract: This study towards the Map-Matching process that is useful to align a location of Global Positioning System (GPS) of vehicles on the digital road networks. Today's GPS-enabled vehicles in developed countries generate a big volume of GPS data. On the other hand, the development of new roads in the city enables the road network very complex and difficult to match the vehicles' location. So therefore, different techniques (i.e., pre-processing techniques) may be applied before the map-matching process is a recent concern of the Intelligent Transport System (ITS) research community. In this paper, we introduce the pre-processing technique; splitting the road network graph and processing the Single Source Shortest Path (SSSP) in synchronize parallel processing in the Hadoop environment. The proposed technique enables the map-matching schemes efficient to align the GPS points on the digital road networks. In the experimental work, the results of the map-matching schemes (i.e., found in the literature review) incorporated with our proposed pre-processing technique shows better performance in aspect to the response time.

Keywords: Map-Matching, GPS, splitting a road network, Pre-processing, SSSP, Bulk synchronize parallel,BSP

1. INTRODUCTION

A fundamental pre-process and useful service in Location-Based Services (LBS) is a map-matching process [1]. Specifically, the map-matching process is required in LBS when aligning the locations of Global Positioning System (GPS) with the road network graph on a given digital map. Several applications for GPS-enabled vehicles such as moving objects management and driving directions use map-matching process as a fundamental step. More specifically, the map-matching process is categorized into three categories: (a) local or incremental method [2, 3], (b) global method [1] [4, 5] and (c) statistical method [6, 7]. In the first category, the algorithms effort to discover the local match of geometry wherein the small portion of the space close to the GPS point is considered. This approach provides the better results in aspect to accuracy with high sampling frequency of the trajectory i.e., 2-5 seconds time interval between GPS points. In the second category, the algorithms consider the complete trajectory for map-matching with a digital road network. The global methods provide the better results in aspect to the accuracy with low-sampling frequency of the trajectory i.e., more than 120 seconds time interval between GPS points. The methods in the third category performs with the statistical methods such as Bayesian classifies, Hidden Markov model [6], Kalman filter and cubic spline interpolation [7] to handle GPS measurement errors. In aspect to the fast response time, the algorithms follow local/incremental method are very fast, while the algorithms follow global method are slow. More concisely, in aspect to the accuracy, the algorithms follow local/incremental method produce low accurate results; while the algorithms follow global method provides high accurate results.

Several LBS applications are deployed online that need a fast result. A map-matching technique therefore is needed which must be deal the low-sampling GPS data and produce fast result. In practice, we found in the well-known map-matching algorithms that the shortest path queries are most frequently used process and most time consuming part of the algorithms.

In this paper, we propose a pre-processing technique for map-matching process, which is approached as follows: (a) The pre-processing technique in the developed model is based on splitting the road network

E-mail: chandio.aftab@usindh.edu.pk, zhangfan@siat.ac.cn, zubairabassi@gmail.com, cskorejo@gmail.com, saleem@usindh.edu.pk

graph for pre-processing the Single Source Shortest Path (SSSP) in Bulk Synchronize Parallel (BSP) processing in the Hadoop environment, i.e., clouds. (b) In the experimental work, the proposed pre-processing technique is incorporated with the map-matching schemes (i.e., Spatial and Temporal MapMatching, ST-MM [1] and Location-Based MapMatching, LB-MM [8] found in the literature review) providing better performance in aspect to overall response time with slightly effects on the accuracy.

The rest of the paper is organized as follows. Section 2 addresses the related work and Section 3 presents the proposed pre-processing model. In the sequel, in Section 4, we provide the experimental results and discussions, while in Section 5, we show the conclusion of the paper.

2. RELATED WORK

Processing of Shortest Path Queries (SPQs) on spatial road network database has been focused recently in the state-of-the-art in order to reduce memory and computation overheads. Wherein spatial road network database, roads are modeled in the graph, where road junctions and road segments are nodes and edges, respectively. To propose new model of road network is fundamental problem in Geographic Information System (GIS) databases for LBS applications [9-13]. The best map-matching approaches are highly depended on the accuracy and correctness of underlying road networks [11]. For instance, Hu et. al. [9] proposed an efficient index to discretize the distances between objects and network nodes into categories, called distance signature for distance computation and query processing over long distances. Kolahdouzan and Shahabi [10] proposed: "a novel approach to evaluate k-Nearest Neighbor (kNN) queries in spatial network databases using first order Voronoi diagram based on partitioning a large network to small Voronoi regions, and then pre-computing distances both within and across the regions". Authors claimed that the process of pre-computing distances and partitioning a large network to small regions can save storage and computation cost. Author in [12] proposed quick map-matching algorithm based on structure of road network, on which road network is divided into two levels and partitions into small grids (mesh). Author claimed that old road network model is not sufficient need to be adapted to the requirement for the efficient map-matching.

Since SPQs (i.e., Dijkstra's algorithm) is only efficient for shortest distances [9], and to minimize the searching and storage cost, the optimal category partitions of spatial road network is needed. By our practical approach and comparative analysis study, we show that the proposed technique of partitions in road network is efficient and robust for SPQs in map-matching.

Main difference between current state-of-the-art and in our work is (a) we pre-compute shortest path queries between all nodes while previous work pre-compute the query between nodes and objects specifically. Furthermore, (b) our system pre-compute the results can be used in different applications which need to perform shortest path query rather in one application they are built for. Common types of queries can also use these results to accelerate the query processing time. Moreover, (c) we eliminate storage overheads as much as possible through splitting road network to small grids and finding redundant data in pre-computed results. (d) We use more efficient technique to pre-compute results, which perform on parallel computing. This technique is not only beneficial for pre-compute results at once but also it is more helpful when road network need to modify then only particular region of the road network will be computed by parallel computers.

3. THE PROPOSED PRE-PROCESSING MODEL

In our proposed model, we split a road network graph into small regions/partitions. The purpose of splitting road network graph into small partitions is to store the results into files in such a way that the required partition can be loaded into the memory. This approach does not only provide an advantage of low-cost storage and efficient query processing but also road network could be well-managed in case of new roads updated by transportation department in the road network database. In the case of updating road network database, only specific partition of the graph would be computed the shortest path distances and spatial information.

A pseudo-code of the proposed technique for partitioning a given road network graph is depicted in Algorithm 1. In this paper, we use the important notations and descriptions which are shown in the Table II.

TABLE II. NOMENCLATURE

Notation	Description
SPQ	Shortest Path Query
BSP	Bulk Synchronous Parallel
$G(V,E)$	A digital road network graph
V	Vertexes of a graph
E	Edges of a graph
K	A number of sub-graphs (G_K)
k	A given number required for partitions
$minC$ and $maxC$	Smallest (minimum) and largest (maximum) value of graph's coordination (i.e., longitude and latitude)
v	Total number of vertexes in a graph
$partialC$	Partial coordination of a partition
TH	Threshold value to adjust the partition coordination
$approxV$	Approximate number of vertexes in a partition
$totalV$	Total number of vertexes counted into the space covered $minC$ and $maxC$ coordination of the partition
$lowC$and $highC$	Coordinate values used for extra space covered in order to connect the neighbor partition.

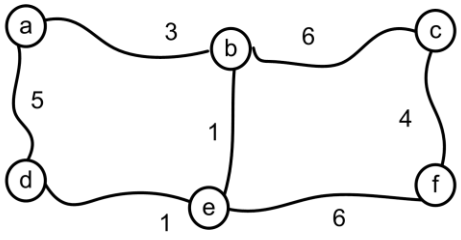
Specifically, this algorithm takes an input of a road network graph $G(V,E)$ and a number k which is given for producing K number of partitions formulated in sub-graphs. An output of this algorithm provides K number of sub-graphs, i.e., $G_K(V,E) = \{G_1(V,E),\dots,G_k(V,E)\}$, where $G_K \epsilon G(V,E)$. Each sub-graph G_k keeps its boundary values such as the maximum $maxC$ and minimum $minC$ longitude and latitude coordinates [12]. In order to balance a load in each sub-graph, our proposed technique guarantees that each sub-graph is created while satisfying a constraint. Such constraint considered in our paper is to create sub-graphs $G_K(V,E)$ with an approximate equal number of nodes. Specifically, to meet the constraint, each sub-graph is adjusted by moving their border's coordinates with a specific value. Initially, this algorithm finds minimum $minC$ and maximum $maxC$ coordinates of the entire graph $G(V,E)$ in respect to find both: (1) a partial coordinates $partialC$ and (2) a moving border value TH to adjust the borders for each partition (line 1-3). The moving border value TH can be further tuned with $percentage$ value. Next, the algorithm finds a number of approximate vertexes $approxV$ in each sub-graph by dividing the number of vertexes v in the entire graph with the number of partitions k (line 5). Then a loop generates K number of sub-graphs (line 6-16). At the beginning of the loop, a total number of vertexes $totalV$ between *left_border* and *right_border* of the current partition will be calculated (line 8). If a difference of (a) total number of vertexes $totalV$ (in the current partition) and (b) approximate vertexes $approxV$ (for each partition) is not greater than total vertexes of an extra space between $lowC$ and $highC$, then the algorithm creates a sub-graph (line 9-12). Otherwise, by TH the algorithm moves a border of the partition to adjust the coordinates of the sub-graph (line 13-15) until to satisfy the constraint (line 9). The output of this algorithm is visualized in Fig. 2, where we set $k = 10$ and it generated $K = 10$ sub-graphs.

Algorithm 1. An algorithm to partition a road network graph

Input: A road network graph $G(V,E)$ and k number for partitions in sub-graphs.
Output: K number of road network sub-graphs $\{G_1(V,E),\dots,G_k(V,E)\}$
$\therefore\ G_K \subseteq G(V,E)$

```
1: find minC and maxC values of the graph G(V,E) coordinates
2: set partialC = (maxC − minC) / k
3: set TH = partialC * percentage
4: set highC = −800
       lowC  = +800
5: set approxV = v / k
   ∴ v is a total number of vertexes in the G(V,E)
6:  set partition = 1
7:  while partition ≤ k do
8:    set totalV = V vertexes between left_border
       and right_border
9:    if (totalV − approxV) ≤ vertexes between lowC
       AND highC then
10:     create a partition G_partition(V,E) sub-graph,
11:   set partition = partition + 1
12:   else
13:     adjust the border of currant partition by
       moving coordination with TH value
14:   end if
15: end while loop
16: return G_1(V,E),...,G_K(V,E)
```

After partitioning of road network graph, the proposed pre-processing technique further generates all-pairs shortest paths between all vertexes in the graph. The above problem has been solved with SSSP function based on Dijkstra algorithm [14]. The SSSP function computes shortest distances from a single source node to all-pairs in the graph. Fig. 1 shows an SSSP example for a six-node network graph connected with seven edges [15]. Each edge is weighted with different value, shown in figure (a). The SSSP results of each node in the graph are revealed into figure (b).



(a)

a	
b	3
c	9
d	5
e	4
f	10

b	
a	3
c	6
d	2
e	1
f	7

c	
a	9
b	6
d	8
e	7
f	4

d	
a	5
b	2
c	8
e	1
f	7

e	
a	4
b	1
c	7
d	1
f	6

f	
a	10
b	7
c	4
d	7
e	6

(b)

Figure. 1 Illustration of an SSSP example: **(a)** network graph and **(b)** SSSP results.

We used parallel computing environment to process the SSSP function on given road network graph. Parallel computing provides an efficient way to process a large graph computation. Specifically, BSP [16-18] parallel computing paradigm implemented in HAMA [19] graph processing tool is selected to process SSSP function.

We modify the SSSP function implemented in Hama according to our problem requirement. In this paper, we used our modified SSSP function approaching BSP parallel computing model proposed in our previous work [15]. Different to our previous work, we consider in this paper to reduce the storage cost by removing redundant data which is highlighted in Fig. 1(b). The complete pseudo-code of the modified SSSP function approaching BSP parallel model is clearly explained in our previous work [15].

4. EXPERIMENTAL RESULTS

A. Setup

In our experiment, we created the Hadoop based computation environment. The Hadoop (version 1.1.2) setup is installed on a total of three physical machines, with the one dedicated for the master node, while the rest ones dedicated for the worker nodes. Each of them is comprised of 8 CPUs Intel(R) Xeon(R) with 2.20GHz speed and 32GB memory. For the pre-processing step, the BSP-based Hama (version 0.6.2) parallel processing technique is employed on the top of the Hadoop environment.

B. GPS Data

- In our experiment, we used the road network graph of Shenzhen city in China. The road network graph is consisted of 86335 vertices and 133397 road segments [20]. Each vertices and road segments are identified by GPS locations stored in the dataset. Fig. 2 visualizes the road network of Shenzhen after applying the proposed partitioning technique.
- Moreover, we used a real-world trajectory dataset contained a number of trajectories collected within 24-hours of a day from a taxicab which was traveled around Shenzhen city in China.
- Furthermore, a synthetic trajectory dataset randomly generated by our own simulator is also used in our experiments. We adopted the method for creating synthetic trajectory dataset from [1].

C. Comparison Performance Metrics

We incorporated our pre-processing technique with the fundamental location-based service called map-matching scheme. The ST-MM [1] and LB-MM [8] map-matching schemes are considered to evaluate the proposed pre-processing technique. We considered two primary performance metrics: **(a)** the percentage of total Correct Matching Points (CMP) in order to analyze accuracy and **(b)** the running time in order to analyze response time. We define the following equation to calculate the percentage of CMP [21].

$$CMP = \frac{correct\ matched\ points}{total\ number\ of\ points\ to\ be\ matched} \times 100\% \qquad (1)$$

Additionally, we investigate the studied map-matching schemes by setting the parameters. Such as Candidate Points (CPs) for each GPS point equals to=5 and Error Circle Radiuses (ECR) equals to=100 meters in the ST-MM. In the LB-MM the minimum value and maximum value of CPs are set 3 and 7, respectively, while the minimum value and maximum value of ECR

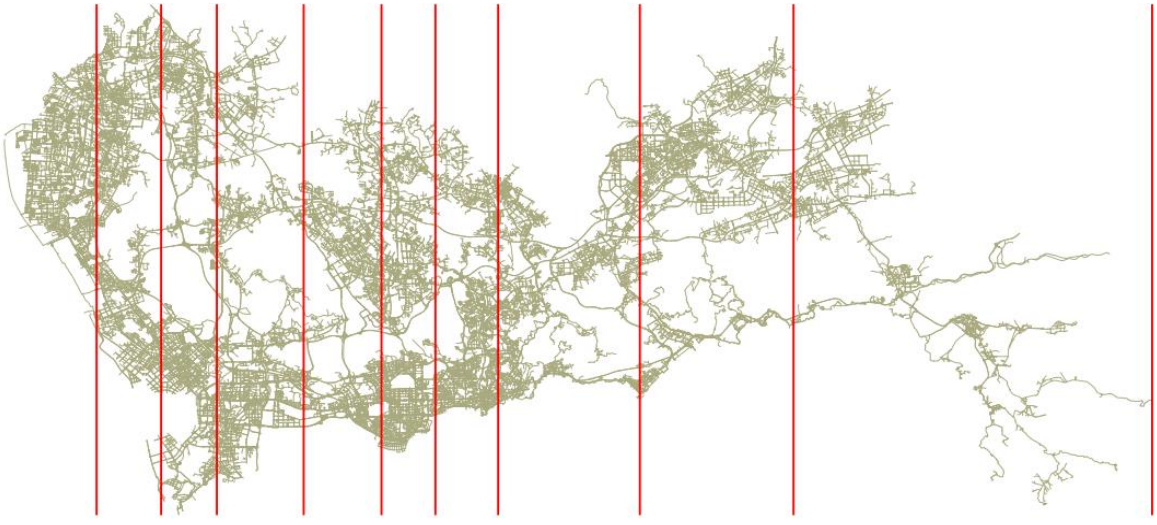


Figure. 2 Visualized road network of Shenzhen after applying partitioning technique

are set 60 and 150 meters, respectively. We call ST-MM-U and LB-MM-U map-matching schemes (i.e., update versions) when applying the proposed pre-processing technique. In order to maintain the fair comparison, we set same parameters values in the studied map-matching schemes (i.e. ST-MM, LB-MM, ST-MM-U, and LB-MM-U). All of the above algorithms are simulated in our own simulator build in Java. Java programing language supports database connectivity [22].

D. Results and Discussions

First of all, we give a complete road network graph as an input to parallel processing environment to pre-process SSSP function. Table I. shows that one node generates 73515 records reserving total 1.76 MB storage space, while all nodes reserve a total of 129.7 GB storage space. It must be noted that nodes means vertexes of the road network graphs.

TABLE I. PRE-PROCESS STEP GENRATES DATA RESERVING THE STORAGE SPACE FOR A COMPLETE ROAD NETWORK GRAPH

Node	Edges	Total cost/edge (bytes)	Storage cost (MB)	Storage cost (GB)
1 Node	73515	24	1.76436	0.001764
73515 Nodes	5404455225	24	129706.9	129.7069

From the table we can interpret that the pre-processing of complete road network (without partitions) is unacceptable. In order to minimize the storage cost, the partition technique of road network graph is required. We split the road network graph to small equal regions/grids, shown in the below table (i.e., Table II.). In the table of graph decomposition, our partition technique distributed graph into ten partitions, each partition comprised an approximate same number of nodes. Each node generated a number of records with equal to the total number of nodes. The pre-compute results for each partition will be stored in separate table in spatial database as map-matching algorithm easily make searching query within time period. In order to minimize the storage cost, we found the redundant data in the output of the pre-processing step, as shown in Fig. 1 (b). When we eliminate un-used data from pre-compute results, then the storage space will be reduced and the searching query will be faster. It is clearly shown in Table II that the filtered data can save an half storage cost.

We further evaluate our pre-processing technique by incorporating with map-matching schemes, i.e., ST-MM-U and LB-MM-U. Fig 3 shows the map-matching results in aspect to the overall running time and the accuracy performance metrics. Fig. 3(a) shows the average running times when varying the number of GPS points per trajectory. The results shown in Fig 3(a) clearly reveal that the map-matching schemes when apply the pre-processing technique (i.e., ST-MM-U and LB-MM-U) provided fast response time. More-precisely, the ST-MM and LB-MM with the proposed pre-processing techniques (i.e., ST-MM-U, LB-MM-U) took much less computation time as compared to basic ST-MM and LB-MM algorithms. Another remark is that the ST-MM-U and the LB-MM-U outperform the ST-MM and LB-MM in aspect to accuracy performance metric. Fig. 3(b) shows the comparison results in terms of accuracy

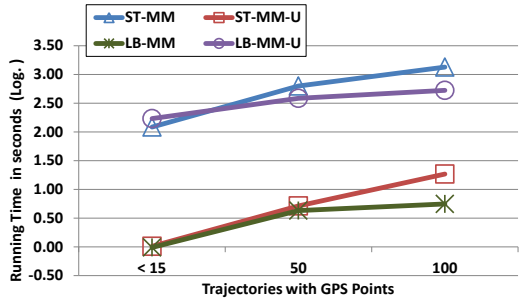
TABLE II. GRAPH DECOMPOSITION: PRE-PROCESS STEP GENRATES DATA RESERVING THE STORAGE SPACE FOR ROAD NETWORK GRAPH PARTITIONS

Partitions No.	Nodes	Overlapping nodes	Un-filtered data			Filtered data		
			KB/Node	Total cost	Storage cost (GB)	KB/Node	Total cost	Storage cost (GB)
1	7803	10781	345	3719445	3.547139	345	1859723	1.77357
2	7803	12835	408	5236680	4.994087	408	2618340	2.497044
3	7528	12393	405	5019165	4.786649	405	2509583	2.393324
4	8140	11996	387	4642452	4.427387	387	2321226	2.213694
5	6968	11278	363	4093914	3.904261	363	2046957	1.95213
6	7803	12240	394	4822560	4.599152	394	2411280	2.299576
7	7738	12455	403	5019365	4.786839	403	2509683	2.39342
8	7771	12615	410	5172150	4.932547	410	2586075	2.466273
9	8016	9979	318	3173322	3.026316	318	1586661	1.513158
10*	3944	4464	87	388368	0.370377	87	194184	0.185188
Total	**73514**	**111036**	**3520**	**41287421**	**39.37475**	**3520**	**20643711**	**19.68738**

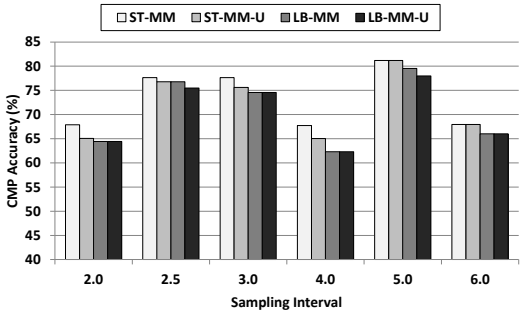
performance metric (i.e., CMP). Specifically, it shows the average CMP of synthetic trajectories when varying the sampling-rate from 2 to 6 minutes per trajectory. It can be seen that our proposed strategies produced almost the same results with the ST-MM and LB-MM algorithms. Another remark is that when the trajectory is travelled on more partitions of the road network then the accuracy of ST-MM-U and LB-MM-U is slightly degraded as compare to the ST-MM and LB-MM.

5. CONCLUSIONS

This paper addresses the proposed pre-processing technique for map-matching strategies. We thoroughly described the pre-processing techniques and their steps used in the map-matching process. Furthermore, the technique is tested and implemented incorporated with the well-known SPQ-based map-matching schemes i.e., ST-MM and LB-MM. The results shown that the studied map-matching schemes provided better results when apply the proposed pre-processing technique.



(a)



(b)

Figure. 3 (a) Running time when varying the number of GPS points in trajectories, **(b)** Accuracy when varying sampling interval (synthetic dataset).

REFERENCES

[1] Lou, Y., et al. *Map-matching for low-sampling-rate GPS trajectories*. in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*. 2009. ACM.

[2] Greenfeld, J.S. *Matching GPS observations to locations on a digital map*. in *National Research Council (US). Transportation Research Board. Meeting (81st: 2002: Washington, DC). Preprint CD-ROM*. 2002.

[3] Wenk, C., R. Salas, and D. Pfoser. *Addressing the need for map-matching speed: Localizing global curve-matching algorithms*. in *Scientific and Statistical Database Management, 2006. 18th International Conference on*. 2006. IEEE.

[4] Alt, H., et al. *Matching planar maps*. in *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*. 2003. Society for Industrial and Applied Mathematics.

[5] Brakatsoulas, S., et al. *On map-matching vehicle tracking data*. in *Proceedings of the 31st international conference on Very large data bases*. 2005. VLDB Endowment.

[6] Pink, O. and B. Hummel. *A statistical approach to map matching using road network geometry, topology and vehicular motion constraints*. in *Intelligent Transportation Systems, 2008. ITSC 2008. 11th International IEEE Conference on*. 2008. IEEE.

[7] Hummel, B. and K. Tischler. *Robust, gps-only map matching: Exploiting vehicle position history, driving restriction information and road network topology in a statistical framework*. in *GIS Research UK Conference (GISRUK)*. 2005.

[8] Chandio, A.A., et al., *An Approach for Map-Matching Strategy of GPS-Trajectories Based on the Locality of Road Networks*, in *Internet of Vehicles - Safe and Intelligent Mobility*, C.-H. Hsu, et al., Editors. 2015, Springer International Publishing. p. 234-246.

[9] Hu, H., D.L. Lee, and V.C.S. Lee, *Distance indexing on road networks*, in *Proceedings of the 32nd international conference on Very large data bases*. 2006, VLDB Endowment: Seoul, Korea. p. 894-905.

[10] Kolahdouzan, M. and C. Shahabi, *Voronoi-based K nearest neighbor search for spatial network databases*, in *Proceedings of the Thirtieth international conference on Very large data bases - Volume 30*. 2004, VLDB Endowment: Toronto, Canada. p. 840-851.

[11] Liu, K., et al., *Effective map-matching on the most simplified road network*, in *Proceedings of the 20th International Conference on Advances in Geographic Information Systems*. 2012, ACM: Redondo Beach, California. p. 609-612.

[12] Zuyun, W., et al. *A quick map-matching algorithm by using grid-based selecting*. in *Education Technology and Training, 2008. and 2008 International Workshop on Geoscience and Remote Sensing. ETT and GRS 2008. International Workshop on*. 2008. IEEE.

[13] Tiwari, S. and S. Kaushik, *Scalable Method for k Optimal Meeting Points (k-OMP) Computation in the Road Network Databases*, in *Databases in Networked Information Systems*. 2013, Springer. p. 277-292.

[14] Dijkstra, E.W., *A note on two problems in connexion with graphs.* Numerische mathematik, 1959. **1**(1): p. 269-271.

[15] CHANDIO, A.A., et al., *Towards adaptable and tunable cloud-based map-matching strategy for GPS trajectories.* Frontiers of Information Technology & Electronic Engineering, 2016.

[16] Kajdanowicz, T., P. Kazienko, and W. Indyk, *Parallel Processing of Large Graphs.* arXiv preprint arXiv:1306.0326, 2013.

[17] Malewicz, G., et al. *Pregel: a system for large-scale graph processing.* in *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. 2010. ACM.

[18] Chandio, A.A., N. Tziritas, and C.-Z. Xu, *Big-Data Processing Techniques and Their Challenges in Transport Domain.* ZTE Communications, 2015. **13**(1): p. 50-59.

[19] Seo, S., et al. *Hama: An efficient matrix computation with the mapreduce framework.* in *Cloud Computing Technology and Science (CloudCom), 2010 IEEE Second International Conference on*. 2010. IEEE.

[20] Chandio, A.A., F. Zhang, and T.D. Memon, *Study on LBS for Characterization and Analysis of Big Data Benchmarks.* Mehran University Research Journal of Engineering and Technology, 2014. **33**(4): p. 432-440.

[21] Yuan, J., et al., *An interactive-voting based map matching algorithm*, in *Mobile Data Management (MDM), 2010 Eleventh International Conference on*. 2010, IEEE. p. 43-52.

[22] Chandio, A.A., et al., *An Implementation of Web Services for Inter-Connectivity of Information Systems.* Int. J. Com. Dig. Sys, 2014. **3**(3): p. 219-225.

Aftab Ahmed Chandio is currently working as a Lecturer of Institute of Mathematics and Computer Science in University of Sindh, Jamshoro, Pakistan, since January 2007. Dr. Chandio received his PhD, doctor of engineering degree in Computer Applied Technology from University of Chinese Academy of Sciences, Beijing, China and engaged in PhD research in Center for Cloud Computing Research of Shenzhen Institutes of Advanced Technology (SIAT) Chinese Academy of Sciences (CAS), Shenzhen, China, from July 2011 to January 2016. Dr. Chandio obtained his BS (Hons) degree in Computer Science from University of Sindh, Jamshoro Pakistan, from January 2003 to December 2006. His research interests include cloud computing, big data, distributed computing, resource management, job scheduling strategies, data center energy efficiency, workload characterization, and map-matching GPS trajectories. His research work appears in over 16 publications in journals and conferences, including Springer's Cluster Computing and FITEE, ZTE Communications, IJCDS, IEEE/ACM ISPA, IEEE ICARCV, LNCS, LNECS. He also served as a Session Chair in IEEE 11th ISPA in Melbourne Australia and Springer's 2nd IOV in Chengdu China. He has 'STOOD THIRD' in BS (Hons) in Computer Science. He was a recipient of the 'Dean Merit Scholarship' awards of SIAT CAS for 2012 as well as for 2015.

Fan Zhang is currently an Associate Professor with Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen China. Dr. Zhang received the Ph.D. degree in communication and information system from Huazhong University of Science and Technology in 2007. From 2009 to 2011, he was a Postdoctoral Fellow with the University of New Mexico and University of Nebraska-Lincoln. His research interests include big data processing, data privacy, and urban computing.

Zubair Ahmed Kalhoro is a Lecturer of Institute of Mathematics and Computer Science in University of Sindh, Jamshoro, Pakistan. Dr. Kalhoro received his PhD in Computational Mathematics from College of Mathematical Sciences, Xiamen University, Xiamen China from 2012 to 2016 and MSc in Mathematics from Institute of Mathematics and Computer Science, University of Sindh, Jamshoro Pakistan in 2005. His research interests include applied numerical linear algebra and matrix equations.

Imtiaz Ali Korejo is currently working as an Associate Professor in the Institute of Mathematics and Computer Science, University of Sindh, Jamshoro, Pakistan since April 2012. He received his Ph.D. from the Department of Computer Science, University of Leicester, United Kingdom in 2012. Dr. Imtiaz Ali Korejo received his B.Sc. (Hons) and M.Sc.(Hons) in Computer Science from University of Sindh, Jamshoro, Pakistan, in 1999, and 2000, respectively. He worked as a research associate in the Institute of Mathematics and Computer Science, University of Sindh, Jamshoro, Pakistan from 2001 to April 2003, as a Lecturer in the same institute from April 2003 to 2012; and Assistant Professor in the Institute of Mathematics and Computer Science. His research interests are evolutionary algorithms, genetic algorithms and adaptive approaches.

Muhammad Saleem Chandio is a Professor and Director of Institute of Mathematics and Computer Science in University of Sindh, Jamshoro, Pakistan. Professor Dr. Chandio received his PhD from Department of Computer Science, University of Wales, Swansea, Wales, UK, in 2002 and MSc in Mathematics from Institute of Mathematics and Computer Science, University of Sindh, Jamshoro Pakistan in 1986.