A Map-Matching Aware Framework For Road Network Compression

Abdeltawab M. Hendawi[1,2] Amruta Khot[1] Aqeel Rustum[1]
Anas Basalamah[3] Ankur Teredesai[1] Mohamed Ali[1]

[1]*Center for Data Science, Institute of Technology, University of Washington Tacoma, WA USA*
[1]{hendawi, akhot, binrusas, ankurt, mhali}@uw.edu

[2]*Department of Computer Science and Engineering, University of Minnesota, Twin Cities, MN, USA*
[2]{hendawi}@cs.umn.edu

[3] *Computer Engineering Department and KACST GIS Technology Innovation Center Umm Al-Qura University, Makkah, Saudi Arabia*
[3]ambasalamah@uqu.edu.sa

***Abstract*—We demonstrate a novel location aware services framework termed COMA for efficient compression and map-matching of road-network graph data. Key innovations include working demonstration of a new compression algorithm to eliminate nodes and edges that do not affect graph connectivity while ensuring object location map-matching accuracy. The demonstration features: (1) Algorithm to leverage compressed versions of road-network graphs for map-matching of objects locations to correct road edges *w*ithout decompression; (2) Upwards of 75% compression ratio implying significant savings for road-network data transmission costs; a key constraint for internet of things (IOT) devices, and (3) Use of a new controllable parameter; termed conflict factor $\mathcal{C}$, whereby location aware services can trade the compression efficiency with map-matching accuracy at varying granularity. In addition to above features the demonstration features an extensible framework that enables experimentation and comparison between various compression and map-matching algorithms in a rich interactive interface. In this paper we outline data management challenges for location aware services using various scenarios for compression of a real road-network map of a large region of United States, along with both real and synthetic moving object trajectories distributed over this map. We describe the COMA framework through its map-based Graphical User Interface, ability to select the area of interest from the road-network map, and submit a compression request. COMA can export and save the compact versions of the selected area in different formats and plot the compressed graph over the original map for visual inspection to study the differences between compressed and uncompressed versions and various related statistics.**

I. Introduction

Context awareness is extremely important for the next generation Internet of Things (IOT). *L*ocation Awareness is central to context. While we are witnessing unprecedented growth in data management techniques for hand-held and wearable device technology, location aware services are still primitive for IOT platforms and services. Current cloud-based location aware services rely on transmitting underlying map data and various routing algorithms for navigation based triggers. Hence, enhancing both the data representation as well as routing capabilities of such spatial database systems is of significant interest to the data management community. Consider the following three related and very important constraints that cause performance issues in IOT location awareness scenarios. First, since form-factor dominates performance issues, the road-network graph (or a representation thereof) needs to be considerably small(er) to ensure that it occupies as less space as possible to fit into expensive flash memory. Second, due to battery power limitations map transmission and route computing need to be efficient. Then third comes the need to use compacted versions of the road-network map graph with highly accurate map-matching of objects to their locations without the need to compresses and decompress the road-network map repeatedly. As we may assume, solving all three issues promises to significantly advance the state of the art in location awareness for IOT scenarios.

There are in fact several existing compression techniques proposed to help address the representation problem [1], [6], [8], [7], [11], [12]. While these techniques do strive for a high compression ratio to address major performance issues, none of these techniques focus on the quality of map-matching accuracy on the compressed map representation as an optimization parameter. We address this shortcoming in our work. Moreover, the compressed map generated by several prior techniques cannot be used directly to perform map-matching without explicit decompression to restore the original form of the map. This is the second limitation we demonstrably address in our work. While out of scope for the current demonstration our experiments do indicate that we address CPU power savings to increase the battery life and are able to perform map-matching on smaller memory (detailed experimental analysis is covered in a separate manuscript currently under submission).

Interestingly, with lossy compression techniques, the compressed version of the road-network map is not representation equivalent of the original one. Some of the map details are lost during the compression process. Typically the quality of lossy compression techniques are evaluated based on the visual similarity or dissimilarity between the generated map (after compression) and the original version of the map (before compression). While visual similarity may be a valid measure of performance in some applications, we believe that the quality of map-matching using the compressed version of the map is a better measure for next generation of automated location aware services since such a measure can be automatically evaluated rather than relying on subjective human judgement.

In this paper, we present a novel compression framework, named *COMA*, that can significantly reduce the size of a given road-network graph without losing any critical information that might prevent map-matching process on the compacted road-network map.

The main idea of the technique is to selectively get rid of as many nodes as possible from the original road network without encoding the remaining nodes. Such selective node removal also ensures that the affected edges will not cause a conflict with other nearby edges when we do map-matching for object location. A distance threshold parameter, termed *conflict factor* $\mathcal{C}$, determines the ratio of the distance between the original node and the closest conflict edge to the distance between the to-be-added edge (after removing a node) and the closest conflict edge. In this way setting the *conflict factor* controls the compression ratio while preserving the desired accuracy of map-matching on the compressed map data. The larger the value of $\mathcal{C}$, the more compressed is the map, and lesser the accuracy of map-matching. Recall that while it may seem that other compression techniques also work on the same principle, the distinction is that the demonstrated framework is the only one that enables tuning accuracy on compressed data. Consider the example we intend to show where COMA is able to compress a detailed road-network map of Washington State, USA, of 2,817,973 nodes and 5,839,924 edges in about five minutes of CPU time on a regular workstation for various values of conflict factor and map-matching accuracy.

The overall COMA framework and its two main components, namely the *Compression Module*, and the *Map-Matching Module* are shown in Figure 1. In addition, we depict the employed spatial-index. Users interact with the system through its map-based web interface to submit compression and map-matching request, and also to set the system settings. Once the COMA framework receives a compression request it sends it to the compression module which runs on the selected area on the map. The R-tree spatial index is visited next to fetch those parts of the road map that intersect with the given area of interest. If the user selects to test the map-matching accuracy on the produced compact graph, the system launches the map-matching module which in turns accesses the spatial index to retrieve a set of objects trajectories and tries to match them to their corresponding roads and compare the matching correctness against the already known results.

During the demonstration, users interact with COMA to: (1) Submit a compression request as a rectangular region of any area on the road network map of the whole Unites States. (2) Run a map-matching test on the resultant compacted graph. (3) Try different values for *conflict factor* $\mathcal{C}$ to obtain the required compression ratio and map-matching accuracy combination. (4) Export the compacted version of the road network graph of the selected area in CSV and KML formats, and (5) Visualize the resultant graph a layer on Google Maps where the base layer represents the original graph.

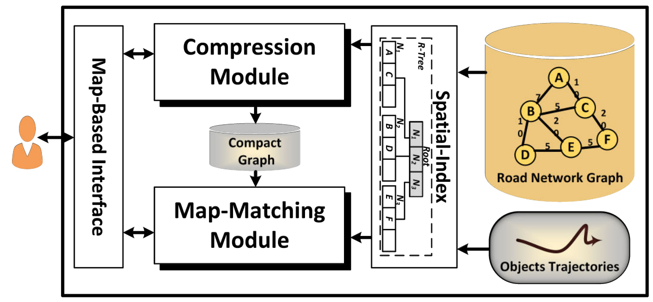


Fig. 1. The demonstrated COMA framework consists of a map based UI, compression and map-matching modules, and a spatial-index each acting in sync with user or system specified conflict factor parameters to facilitate compression with map-matching for object trajectories.

II. COMA Framework

The road compression algorithm and map-matching components are described next.

A. *Compression Module*

The compression algorithm reduces number of nodes and edges in a given road-network graph such that the deletion of node/edge will not cause map-matching ambiguity. Multiple edges can be compressed and represented by a single *bridge edge*. Our aim is to design a compression algorithm that optimizes for minimal false positives and false negatives. We begin by the compression process by selecting an arbitrary node at the given road network graph. To decide if the start node n_v qualifies as a candidate node for deletion (victimized node), we apply two major checks.

First, we check that the deletion of n_v is safe from the graph connectivity perspective. This is done by assuring that the node n_v falls into one of the following cases. (a) It has just two different connected edges, n_i and n_j and both are bi-directional or one of them is an input to n_v and the other is an output from n_v. (b) It is connected to many edges with; exact one input edge and the rest are output edges, or exact one output and the rest are inputs. In this case, all the connected edges have to be unidirectional and none of them is bidirectional.

Second, we check that the deletion of n_v is safe from the map-matching perspective. To achieve this, we examine the newly formed bridge edge $e_{i,j}(n_i, n_j)$, (resulting from connecting the two far ends, n_i and n_j of the input and output edges of n_v). If (1) the candidate bridge edge is closer to the candidate node n_v than any other edge in the vicinity and (2) if the to-be-deleted edges are the closest to the bridge edge, the node n_v is victimized and the new bridge edge replaces the edges of n_v in the graph.

To control the behavior of the compression algorithm, we define a tuning parameter, called the *conflict factor threshold* $\mathcal{C}$.

The conflict factor of a candidate victim node n_v is the distance from this node n_v to the to-be-added bridging edge relative the distance from n_v to the nearest edge in the vicinity. If the conflict factor of node n_v is below the specified conflict factor threshold $\mathcal{C}$, the victimization may take place. Otherwise, the victimization step stops, and no compression is achieved at that node. By leveraging $\mathcal{C}$, we can control the trade off between the compression ratio and the map-matching quality. The higher $\mathcal{C}$ is, the higher the compression ratio we get, and the less the quality of map-matching we guarantee, and vice versa.

B. The Map-Matching Module

Map matching refers to the process of linking a series of GPS locations to their corresponding road segments in the underlying road network graph [9]. Map-matching helps us to know precisely the correct edge on which the object, (vehicle or person) is currently traveling on. Hence, we can accurately answer spatial queries, e.g., finding shortest path or finding nearest point of interest.

The accuracy of the map matching output is highly dependent on the quality of the underlying road-network and should provide a good representation of the real streets along with their directions, connectivity, layout, and intersections. Since the main goal of our compression approach is to significantly compress the size of the road-network graph while maintaining a high quality road-network representation in the compressed version of the graph. There is a choice of several map-matching algorithms and we utilize the $Passby$ map-matching algorithm [9] in our demonstration. It can be substituted by other map-matching algorithms in the future. This algorithm can work on even the most simplified road networks. It does not have to be fed with each single detail in the road network graph.

III. Demonstration Details and Scenarios

We cover various scenarios of demonstrating the COMA framework. An interactive GUI (graphical user interface) as seen in Figure 2 is the entry point for the user to: (a) submit compression request, (b) test the map-matching quality of the results, (c) save and export the resultant compact graph, (d) run a competitive technique, say, Douglas-Peucker [5] for result validation. The framework can also be reviewed from an algorithmic perspective to appreciate how COMA framework internally works in terms of computation, compression, and map-matching, through interactive charts (pie, bar, and line).

Different programming frameworks implement and integrate the system modules as well as the underlying data structures. For example, both the COMA internal compression technique and *Douglas-Peucker* techniques are implemented in C#, while the *Passby* map-matching technique is implemented in C (source code provided by Lui et. al. [9]). The front-end map-based interface is built using a combination of javascript, HTML, CSS, to leverage the Google Maps API and the D3.js visualization packages.

COMA framework is seeded with real road-network graphs obtained from the OpenStreeMap [10] with assistance of Tareeg framework [2]. To examine the quality of map-matching we use both real as well as synthetic object trajectories. The real trajectories are obtained from different sources including the volunteered GPS-traces on OpenStreetMap and the sets provided by the GIS CUP 2012 [3] for the state of Washington, USA. The synthetic data of moving objects is generated using the Brinkhoff's generator [4].

Demonstrated scenarios are:

A. Scenario 1: Submit Compression Request

We interact with the COMA framework to submit a road-network compression request by defining a rectangular area on the rendered map as can be seen in Figure 2(a). We can also directly go to our area of interest on the map if we know the latitude and longitude information, or we pan-and-zoom till the target area becomes visible in the map window. Once selected and on submission, a compression request is sent to the service api. Upon such request, the spatial index, currently an R-tree, is accessed to retrieve the appropriate overlapping region of graph partition (nodes and edges). Then compression module executes the COMA algorithm over the given region. Simultaneously, a progress bar indicates the percentage of compression task completion.

B. Scenario 2: Exporting Compressed Road Network Data

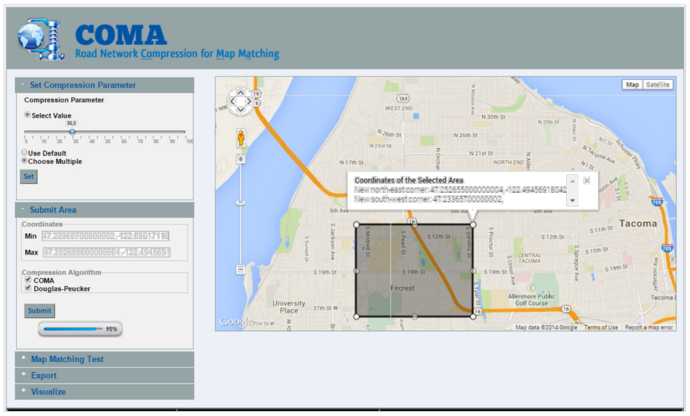
As seen in Figure 2(b), COMA framework supports the ability to export the compressed road network graph in standard formats: (1) The KML, Keyhole Markup Language format supported by Google that is parsed by many spatial visualization engines including Google Earth and (2) CSV (Comma Separated Values) format that is easy to programmatically manipulate, process, and transfer. The resultant compact graph is split and stored in three files, Node <nodeId, Latitude, Longitude>, Edge <edgeId, startNode, endNode, cost>, Edge_Geometry <edgeId, list of Latitudes and Longitude of intermediate nodes>. This list of intermediate nodes represents the curvy shape of the bridge edge.

C. Scenario 3: Validating against Competitive Compression Techniques

In this scenario, we are able to run competitive compression algorithms such as the Douglas-Peucker compression technique. Other techniques can be supported in the framework in future. Select the desired check-boxes as depicted in Figure 2(a) and the request will be processed using the two supported techniques. Next compare the outcomes from COMA framework versus Douglas-Peucker, and review the results of different flavors of compression.

D. Scenario 4: Map-Matching Test

Now that the compression is done, we do a map-matching test to examine the accuracy. Recall that we do map-matching on compressed graph and don't have to decompress the road-network. In most cases, the compressed graph produced by COMA can achieve a very high map-matching accuracy (close to 99%). However, in some cases when the graph is highly



Set Compression Parameter

Compression Parameter

(●) Select Value 30,0

( ) Use Default

(●) Choose Multiple

Set

Submit Area

Coordinates

Min 47.23365700000002,-122.53017118

Max 47.25265500000004,-122.4945691

Compression Algorithm

☑ COMA

☑ Douglas-Peucker

Submit

95%

Map Matching Test

Export

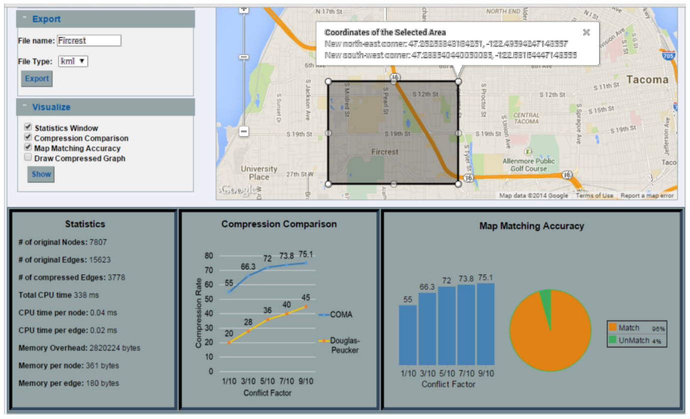
Visualize

Coordinates of the Selected Area

New north-east corner: 47.252655000000004,-122.49456918042

New south-west corner: 47.23365700000002,

(a) Compression Request And Parameters



Export

File name: Fircrest

File Type: kml

Export

Visualize

☑ Statistics Window

☑ Compression Comparison

☑ Map Matching Accuracy

☐ Draw Compressed Graph

Show

Coordinates of the Selected Area

New north-east corner: 47.25253843184251, -122.49394247143557

New south-west corner: 47.23354044005008, -122.53154447143555

Statistics

# of original Nodes: 7807

# of original Edges: 15623

# of compressed Edges: 3778

Total CPU time 338 ms

CPU time per node: 0.04 ms

CPU time per edge: 0.02 ms

Memory Overhead: 2820224 bytes

Memory per node: 361 bytes

Memory per edge: 180 bytes

(b) Results Visualization And Export

Fig. 2. The Graphical User Interface in The COMA System

dense, or an aggressive conflict factor is set, or the nature of vicinity, e.g., forest areas that negatively affect the precision of the GPS readings, a prefect map-matching might not be feasible. For the demonstration we use the *Passby* map-matching algorithm to ensure high quality map-matching even with minimal detailed road networks and low sampling moving objects trajectories [9]. Currently, we leverage the sets of objects' trajectories that we have already collected from different sources as described earlier. However, we are extending the framework to upload own trajectories in the selected area to be used for testing the map-matching accuracy in that area.

E. Scenario 5: Managing System Settings (conflict factor)

We have the ability to set the value for the conflict factor $\mathcal{C}$ and study its effects on accuracy and compression. Figure 2(a) indicates how by sliding the tracker bar to the desired $\mathcal{C}$ value and selecting the check box named "$UseDefault$", or by selecting the check box named "$ChooseMultiple$" we can change these values. The last option can invoke the compression module iteratively on several possible preset $\mathcal{C}$ values: 0.1, 0.3, 0.5, 0.7, and 0.9. The middle option runs the compression process using $\mathcal{C} = 0.3$ which we find to be the best value for efficiency and accuracy on the Washington State road-network.

F. Scenario 6: Visualizing The Results

COMA framework has a visualizer component that is quite versatile. It offers the following services: (a) The original road network will be visible as the base layer. Drawing as second layer over the base layer, the output compressed graph with remaining nodes and edges. Ability to add a third layer if we prefer to draw the Douglas-Peucker compression graph for visual comparison and analysis of results. (b) Plotting the efficiency (CPU utilization), and the quality (map-matching accuracy) measurements of recent compression requests through various charts (pie, bar and line) supported through d3.js libary. Examples of such functionality are visible in the right bottom panel of Figure 2(b) representing a histogram for compression ratio variation with $\mathcal{C}$ value changes for selected map area. The line charts in the middle bottom compares the two compression techniques, while the small window in the left bottom provides statistics about CPU and memory overhead consumed to accomplish the submitted task.

References

[1] Alexander Akimov, Alexander Kolesnikov, and Pasi Franti. Reference line approach for vector data compression. In *ICIP*, pages 1891–1894, Singapore, October 2004.

[2] Louai Alarabi, Ahmed Eldawy, Rami Alghamdi, and Mohamed F. Mokbel. TAREEQ: A MapReduce-Based Web Service for Extracting Spatial Data from OpenStreetMap. In *SIGMOD*, Utah, USA, June 2014.

[3] Mohamed H. Ali, John Krumm, Travis Rautman, and Ankur Teredesai. ACM SIGSPATIAL GIS cup 2012. In *GIS*, pages 597–600, November 2012.

[4] Thomas Brinkhoff. A Framework for Generating Network-Based Moving Objects. *GeoInformatica*, 6(2):153–180, 2002.

[5] David H. Douglas and Thomas K. Peuker. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica*, 10(2):112–122, 1973.

[6] Suh Jonghyun, Jung Sungwon, Pfeifle Martin, Vo Khoa T, Oswald Marcus, and Reinelt Gerhard. Compression of digital road networks. In *SSTD*, pages 423–440, Massachusetts, USA, July 2007.

[7] Ali Khoshgozaran, Ali Khodaei, Mehdi Sharifzadeh, and Cyrus Shahabi. A hybrid aggregation and compression technique for road network databases. *Knowledge and Information Systems*, 17(3):265–286, 2008.

[8] Amruta Khot, Abdeltawab Hendawi, Raj Katti, Anderson Nascimento, Ankur Teredesai, and Mohamed Ali. Road network compression techniques in spatiotemporal embedded systems: A survey. In *the International ACM SIGSPATIAL Workshop on Geostreaming, IWGS*, Dallas, TX, USA, November 2014.

[9] Kuien Liu, Yaguang Li, Fengcheng He, Jiajie Xu, and Zhiming Ding. Effective map-matching on the most simplified road network. In *GIS*, pages 609–612, Redondo Beach, CA, USA, November 2012.

[10] OSM. OpenStreetMap (OSM). http://openstreetmap.org, January 2014.

[11] Shashi Shekhar, Yan Huang, Judy Djugash, and Changqing Zhou. Vector map compression: a clustering approach. In *GIS*, pages 74–80, VA, USA, November 2002.

[12] Zongyu Zhang. Vector road network compression : a prediction approach. In *ASPRS*, Reno, Nevada, USA, May 2006.