A Demonstration of RASED: A Scalable Dashboard for Monitoring Road Network Updates in OSM

Mashaal Musleh, Mohamed F. Mokbel
Department of Computer Science and Engineering, University of Minnesota, MN, USA
{musle005, mokbel}@umn.edu

***Abstract*—Road network queries (e.g., shortest path, range, and k-NN) hinge on the road network quality, which, unfortunately, suffer from all sorts of inaccuracy. Given that OpenStreetMap (OSM) has been the de facto open-source map for a myriad of widely used applications, this demo presents RASED; a publicly available scalable dashboard to interactively monitor and analyze billions of OSM updates worldwide. RASED provides the necessary infrastructure that is immensely needed by map analyzers to understand and assess the map quality for anywhere in the world, which is a measure of the query accuracy.**

I. Introduction

Though there is extensive research in academia and industry for developing efficient algorithms for a myriad of road network queries (e.g., shortest path [13], range [12], and k-NN [10] queries), all algorithms have the implicit assumption that the underlying map is accurate. Unfortunately, such an assumption is not always true as maps suffer from all sorts of inaccuracy that significantly degrade the query result accuracy, no matter how efficient are the deployed algorithms [7]. In practice, such map inaccuracy costs delivery companies in US $6 Billion annually, which is mainly due to the inaccuracy of their shortest path query result [11]. This triggered a whole area of research of studying the quality of the underlying maps as a means of understanding its impact on the query result accuracy. Almost all of these studies focus on OpenStreetMap (OSM) [8], known as the Wikipedia of maps, which has recently replaced commercial maps in various sectors of academia, government, and industry [1], [5], [6].

Unfortunately, all such quality assessment studies (e.g., [2], [4], [14]), mostly initiated by the transportation community, have very limited scope and scale, where the focus is only to study a certain city or country road network with heavy manual operations. Up to our knowledge, there is no comprehensive global-scale study for the quality of OpenStreetMap (OSM). The main reason is the huge size and continuous updates of OSM, which force researchers to limit their studies to small regions. For example, OSM has 12+ Billion update records in its history, which accounts for around 3TB worth of raw data, and is continuously increasing [9]. This makes it hard for non-experts to interactively handle such amounts of data and provide insightful quality assessment analysis.

This work is supported by the National Science Foundation, USA, under Grant IIS-1907855.

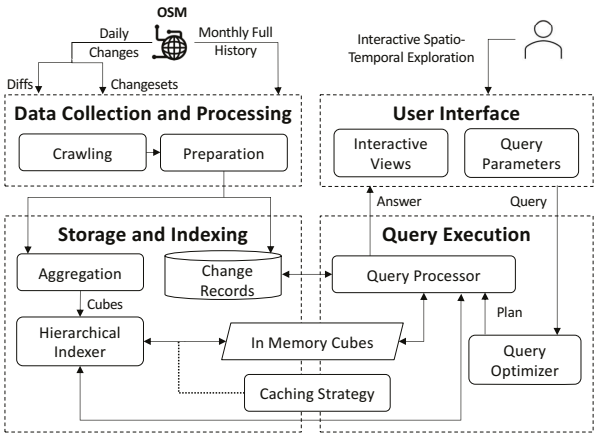

Fig. 1. RASED Architecture

In this demo, we introduce RASED [1]; a publicly available and scalable dashboard to interactively monitor and analyze all OSM road network updates worldwide. RASED provides the necessary infrastructure immensely needed by map analyzers to understand and assess the map quality for anywhere in the world. RASED tackles the scalability challenge through employing: (1) *Aggregation*, which exploits the historical property of map updates and performs offline computations to produce a giant high-dimensional data cube [3], (2) *Slicing*, which cuts the resulted cube into smaller ones that are easily managed in memory, (3) *Hierarchical Indexing*, which stores and retrieves these slices efficiently from the disk, and (4) *Caching*, which utilizes the system available memory by prefetching certain data cubes from the disk. When combined, these techniques provide highly interactive analysis for various sorts of queries over the history of OSM updates since its inception in 2004.

Users of RASED, and conference audience, will be able to interact live with the system to query and visualize various statistics about the map, including number and percentage of OSM updates for each country, comparison between countries, types of updated roads, temporal evolution of the updates, and map metadata availability. All queries to the dashboard can have several filters including temporal ones (e.g., time of update), spatial ones (e.g., country or state), map elements (e.g., road type), and kind of updates, which would all give a better understanding of map status globally.

[1] https://rased.cs.umn.edu

2375-026X/22/$31.00 ©2022 IEEE
DOI 10.1109/ICDE53745.2022.00284

II. RASED Architecture

Figure 1 depicts the architecture of RASED, composed of the following four main modules:

Data Collection and Processing. This module is responsible for continuously crawling OSM updates and preparing them for consumption. In particular, it crawls both the OSM daily update file that only includes the list of updated items and the OSM monthly full history that includes the previous state of each single change over the month. Then, it deploys various preparation tasks, including filtering out irrelevant data, merging multiple datasets, and extracting location information, before sending the data to the *Storage and Indexing* module.

Storage and Indexing. This module takes the prepared data as input, aggregates the data into data cubes [3], and organizes the data in a hierarchical index structure for an efficient retrieval by the *Query Execution* module. Details are in Section III.

Query Execution. This module receives a query from the *User Interface* module and decides the best plan to execute it through an interaction with the *Storage and Indexing* module. Details are in Section IV.

User Interface. This module is user facing, where it receives interactive queries form RASED users, sends them to the *Query Execution* module, gets the results, and generates several views that show the data from different perspectives. Details will be shown through the demo scenarios in Section V.

III. Storage and Indexing

This module is responsible for efficiently storing and retrieving the prepared data through the following:

Precomputed Aggregation. To significantly reduce the response time of RASED queries and make it more interactive, all OSM updates are pre-aggregated in a giant five-dimensional data cube [3], representing temporal (e.g., daily), spatial (e.g., country or region), road/feature type (e.g., residential, service, motor way), map elements (e.g., node, way), and update type (e.g., new, updated) dimensions. To support a variety of analysis queries, RASED, so far maintains such cube for 6,000+ days, 300+ spatial regions include all countries and some selected regions (e.g., US states), 150 road/feature types, three map elements and four kinds of updates. All together makes 3+ billion data cube which would account for 25+ GB. Our aggregated pre-computations maintain such giant data cube offline, with: (a) additions of 500+K daily new entries, and (b) monthly updates of 12+ million entries based on the monthly history updates from OSM.

Hierarchical Temporal Indexing. The increasingly large size of our data cube makes it hard to store it entirely in memory. Meanwhile, directly querying our disk-based data cube will not make our analysis interactive enough for RASED users. Hence, RASED maintains its own hierarchical temporal index (depicted in Figure 2) on top of its data cube by: (a) Slicing the data cube into daily slices, where each day, one slice is added, and (b) Maintaining a hierarchical aggregate index of the daily slices. Sliced daily cubes would be inefficient in answering queries over extended periods as this would require loading a large number of cubes from disk. To overcome this, we employ

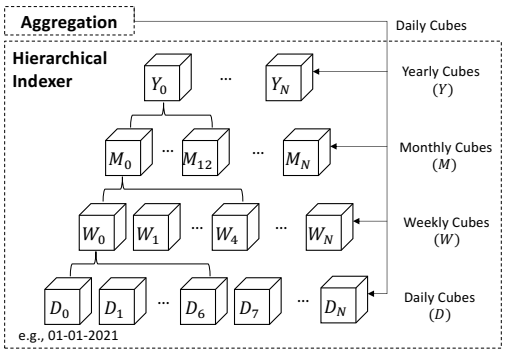

Fig. 2. Hierarchical Index for Data Cubes

a hierarchical index of four temporal levels: daily, weekly, monthly, and yearly. The lower daily level is composed of the daily cubes. The weekly level includes an aggregation of all the seven cubes that are in the same week, and so on for monthly and yearly cubes. It is important to note that all cubes are of the same size, regardless of their temporal level.

Caching Strategy. Given the large number of cubes at all levels of the hierarchical index, it becomes important to smartly preload some of these cubes into memory, which would allow for interactive analysis. Given N available memory slots and the sets Y, M, W, and D of yearly, monthly, weekly, and daily cubes, we preload the following cubes into memory: $\{D_{|D|-i}\}_{i=0}^{\alpha N} \cup \{W_{|W|-i}\}_{i=0}^{\beta N} \cup \{M_{|M|-i}\}_{i=0}^{\gamma N} \cup \{Y_{|Y|-i}\}_{i=0}^{\theta N}$ where α, β, γ, and θ: (a) present the ratio of the N memory slots that will be allocated to each temporal level, (b) have a total sum of 1, and (c) provide a trade-off between aggregation granularity and time coverage, e.g., higher α would cache more details but less covered period. It is important to note that cubes of each level are selected in reverse chronological order as queries are more likely to be about recent updates.

IV. Query Execution

This module is responsible for efficiently executing user queries through the following two main operations:

Query Optimizer. The objective is to find the best query plan that would minimize the number of accessed disk-based data cube for a given user query. There may be alternative plans that would partially or totally answer the query from memory or disk cubes, and from several levels in the hierarchical index. For example, a query about number of updates in USA on Oct 2021 can be answered from either: (a) 31 daily cubes, (b) four weekly cubes and three daily ones, or (c) a single monthly cube. The query optimizer would also take into account which cubes are in memory to come up with the best query plan.

Query Processor. This query processor executes the query plan generated by the query optimizer and loads the selected cubes into memory in the form of Data Frames. The query processor would then perform any remaining computations on the fly using the data frame aggregation/filtration APIs. For queries that request individual samples of map updates, we translate them into SQL and send them to the standard DBMS in the Storage module. Once all computations are done, the query processor sends the results back to users.

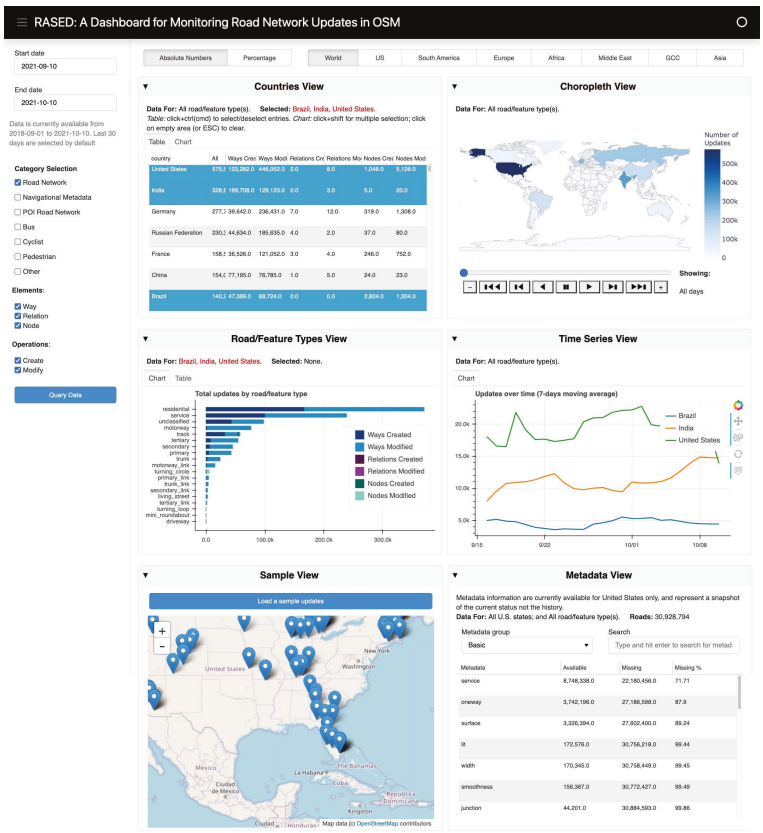

Fig. 3. User Interface of RASED (https://rased.cs.umn.edu)

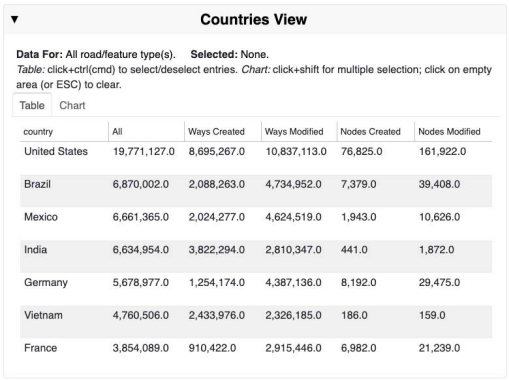

Fig. 4. Country View

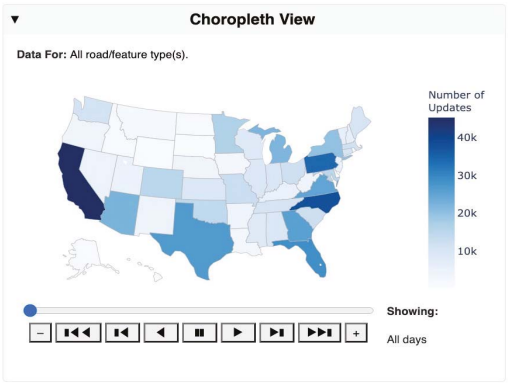

Fig. 5. Choropleth View

V. Demo Scenarios

This section lists several scenarios through which conference audience can interact with RASED to examine its scalability, interactive and comprehensive analysis, and navigate through all map updates worldwide. In particular, the interactive analysis makes the system highly engaging where audience would not feel about the huge scale of data that RASED maintains. Among the numerous possible scenarios that users can interact with RASED, we only list a few below:

Scenario 1: Scalable Query Execution. Figure 3 presents the landing page of RASED where users can specify the following query parameters: (1) temporal range to specify the dates of interest, (2) road types of interest (e.g., road networks, bus or cyclists routes), (3) type of OSM data (e.g., nodes, ways, or relations), and (4) nature of updates (e.g., create or modify). Furthermore, users can customize the view of the query results using two main options: (a) absolute/percentage statistics, which is crucial to see the amount of map updates relative to road network size of each country, and (b) the geographical regions to focus on, which could be the whole world or a specific region such as Europe. Conference audience will be able to see that no matter how large is the query temporal range or spatial region, the query response and all visualizations are highly interactive. This is mainly due to RASED efficient storage and querying techniques.

Scenario 2: Statistics per Country. Figure 4 depicts the Table view of all updates per country. Users can toggle this view as Table or Chart, and can sort entries based on the number/percentage of road changes per country. This helps road network analyzers to understand which countries have more changes in their road network either as an absolute number or percentage, and according to the query parameters (temporal range, road type, and update type) set in Scenario 1. The Chart view of this scenario is categorized by both the road type and nature of updates. Finally, users can use this view as an additional parameter to filter the results of all other scenarios. For example, one can click on one or more countries in the Table view, which will limit all other scenarios to show only the statistics of the selected countries.

Scenario 3: Time-lapse Choropleth View. Figure 5 gives a bird view of all map updates according to the query parameters and geographic region selected in Scenario 1. The view is presented as a Choropleth chart, where users can run a time-laps video of it, which helps in understanding the evolution of road network updates through the temporal range of interest.

Scenario 4: Statistics per Road/Feature Type: Figure 6 shows the Chart view for the update activity for every single road type (e.g., service and residential roads) for the query parameters set in Scenario 1 and the countries selected in Scenario 2. Users can toggle this view as Table or Chart, and

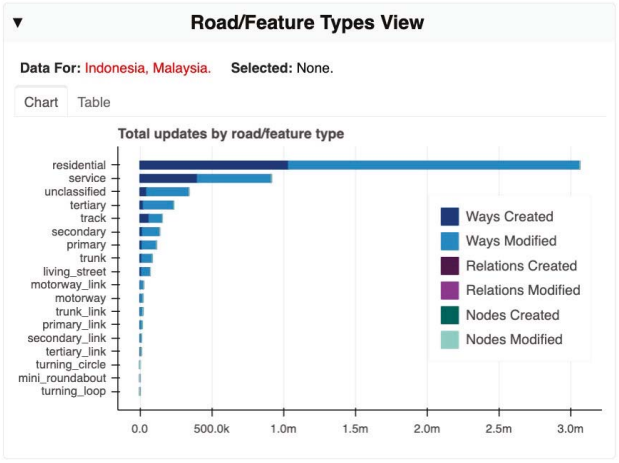

Fig. 6. Road/Feature Types View

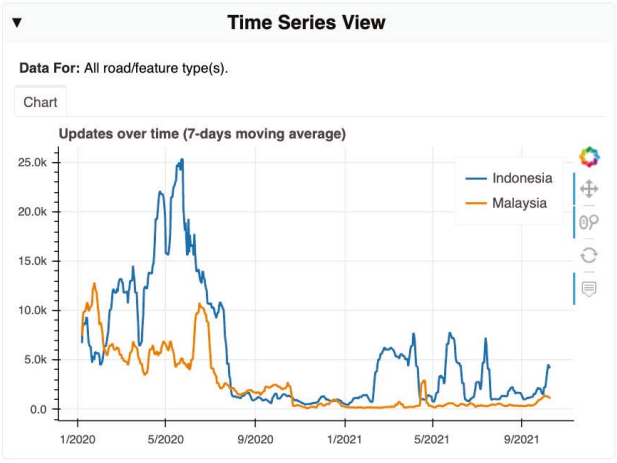

Fig. 7. Time Series View

can sort entries based on various statistics. The Bar Chart view is split into different colors indicating different map elements and different types of updates. This helps map analyzers understand which roads are being updated more than others in which country. In the Table view, selecting one or more road types will reflect on all other scenarios as filters.

Scenario 5: Comparison between Countries: Figure 7 shows a scenario where users can compare the road network update activities between multiple countries per certain query parameters and road/features types. Users can continue to add/remove countries or scroll to zoom in/out of the chart to inspect the values at different points of the timeline. Users will be able to witness the system scalability through the highly interactive analysis over the large scale data maintained by RASED.

Scenario 6: Inspection of Sample Updates: With the Sample View panel, users can retrieve a sample of map updates that satisfy the parameters set in all other scenarios combined. The samples are plotted on the map as pins showing their locations. Clicking on any of these pins would show the "before" and "after" status of this specific update.

Scenario 7: Metadata Completeness Check: Figure 8 depicts a scenario where users can check the percentage of roads annotated with certain metadata (e.g., maximum speed, number of lanes) in a tabular view, which can be sorted based on any of the columns. Users can also select certain road types (e.g., primary) or specific regions (e.g., Washington State) from other views to evaluate the coverage percentage only against these selected regions and road types.

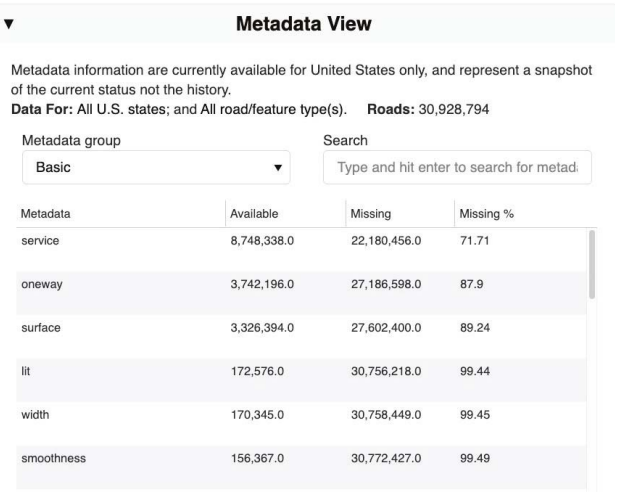

Fig. 8. Metadata View

REFERENCES

[1] Facebook Engineering. MaRS: How Facebook keeps maps current and accurate. https://engineering.fb.com/2019/09/30/ml-applications/mars/.

[2] J.-F. Girres and G. Touya. Quality Assessment of the French OpenStreetMap Dataset. *Trans. in GIS*, 2010.

[3] J. Gray, A. Bosworth, A. Layman, and H. Pirahesh. Data Cube: A Relational Aggregation Operator Generalizing Group-By, Cross-Tab, and Sub-Total. In *ICDE*, 1996.

[4] K. T. Jacobs and S. W. Mitchell. OpenStreetMap Quality Assessment using Unsupervised Machine Learning Methods. *Trans. in GIS*, 2020.

[5] Lyft Engineering. How Lyft discovered OSM is the Freshest Map for Rideshare. https://eng.lyft.com/how-lyft-discovered-openstreetmap-is-the-freshest-map-for-rideshare-a7a41bf92ec.

[6] Money Control. Uber may shun Google Maps for open source ones. https://www.moneycontrol.com/news/business/uber-may-shun-google-maps-for-open-source-ones-report-2764111.html.

[7] M. Musleh, S. Abbar, R. Stanojevic, and M. Mokbel. QARTA: An ML-based System for Accurate Map Services. *PVLDB*, 2021.

[8] OpenStreetMap. http://www.openstreetmap.org/.

[9] OSM Full History. https://planet.openstreetmap.org/planet/full-history.

[10] D. Ouyang, D. Wen, L. Qin, L. Chang, Y. Zhang, and X. Lin. Progressive Top-K Nearest Neighbors Search in Large Road Networks. In *SIGMOD*, 2020.

[11] Traffic Technology Today. Poor maps costing delivery companies US $6bn annually, 2020. https://www.traffictechnologytoday.com/news/mapping/poor-maps-costing-delivery-companies-us6bn-annually.html.

[12] H. Wang and R. Zimmermann. Processing of Continuous Location-Based Range Queries on Moving Objects in Road Networks. *TKDE*, 2011.

[13] L. Wu, X. Xiao, D. Deng, G. Cong, A. D. Zhu, and S. Zhou. Shortest Path and Distance Queries on Road Networks: An Experimental Evaluation. *PVLDB*, 2012.

[14] H. Zhang and J. Malczewski. Accuracy Evaluation of the Canadian OpenStreetMap Road Networks. *International Journal of Geospatial and Environmental Research*, 5(2):1:1–1:14, 2018.