Integration of Mobility Data with Weather Information

Nikolaos Koutroumanis
Department of Digital Systems
University of Piraeus
Greece
koutroumanis@unipi.gr

Georgios M. Santipantakis
Department of Digital Systems
University of Piraeus
Greece
gsant@unipi.gr

Apostolos Glenis
Department of Digital Systems
University of Piraeus
Greece
apostglen46@gmail.com

Christos Doulkeridis
Department of Digital Systems
University of Piraeus
Greece
cdoulk@unipi.gr

George A. Vouros
Department of Digital Systems
University of Piraeus
Greece
georgev@unipi.gr

ABSTRACT

Nowadays, the vast amount of produced mobility data (by sensors, GPS-equipped devices, surveillance networks, radars, etc.) poses new challenges related to mobility analytics. In several application, such as maritime or air-traffic data management, data analysis of mobility data requires weather information related to the movement of objects, as this has significant effect on various characteristics of its trajectory (route, speed, and fuel consumption). Unfortunately, mobility databases do not contain weather information, thus hindering the joint analysis of mobility and weather data. Motivated by this evident need of many real-life applications, in this paper, we develop a system for integrating mobility data with external weather sources. Our system is designed to operate at the level of a spatio-temporal position, and can be used to efficiently produce weather integrated data sets from raw positions of moving objects. Salient features of our approach include operating in an online manner and being reusable across diverse mobility data (urban, maritime, air-traffic). Further, we extend our approach: (a) to handle more complex geometries than simple positions (e.g., associating weather with a 3D sector), and (b) to produce output in RDF, thus generating linked data. We demonstrate the efficiency of our system using experiments on large, real-life data sets.

KEYWORDS

Mobility data, trajectories, weather integration

1 INTRODUCTION

The ever-increasing rate of generation of mobility data by modern applications, including surveillance systems, radars, and GPS-enabled devices, poses new challenges for data management and analysis. To support advanced data analytics, mobility data needs to be enriched by associating spatio-temporal positions of moving objects with external data sources, as the data is ingested. This problem is known as data integration and is particularly challenging in the era of Big Data [2].

One significant data integration task common in all domains, including urban, maritime and air-traffic, is related to weather information. This is due to the fact that weather plays a critical role in the analysis of moving objects' trajectories [3, 8]. The reason is that having available the weather information together with kinematic information enables more complex reasoning about trajectory data, with prominent examples trajectory prediction and clustering. In the former case, the trajectory that will be followed by a moving object clearly depends on weather, while in the latter case common patterns of movement may be revealed when taking weather into account.

Furthermore, our involvement in several EU projects and the interaction with domain experts has strengthened the above observation. Namely, in fleet management use-cases (cf. project Track&Know[1]), fuel consumption can be estimated more accurately if weather information is available (mostly rain-related information). In the maritime domain (cf. projects BigDataStack[2] and datAcron[3]), weather typically affects the trajectory followed by a vessel. Last, but not least, in air-traffic management (cf. project datAcron), storm-related information may affect not only the route of an aircraft, but can also result in regulations for flights and eventually delays, which could probably be predicted. Therefore, a common requirement across all these domains is to have available weather information together with the positions of moving objects.

Unfortunately, despite the significance of integrating mobility data with weather, there is a lack of such publicly available systems or tools that are easy to use. Motivated by this limitation, in this paper, we present the design and implementation of weather integration system, which has several salient features: (a) it works as a standalone and re-usable tool for data integration of mobility data with weather, (b) it is efficient in terms of processing performance, thus making it suitable for application in online scenarios (stream processing), (c) it supports enrichment of complex geometries (e.g., polylines, polygons) with weather data, which is not straightforward.

In summary, we make the following contributions:

- We present a generic system for integrating mobility data represented by spatio-temporal positions with weather information, focusing on ease of use and efficient processing.
- We show how to extend the basic mechanism to perform weather integration for more complex geometries, such as large 3D sectors, which is not straightforward.
- We demonstrate the efficiency of our system by means of empirical evaluation on real-life mobility data sets from different domains (urban, maritime and air-traffic).

The remainder of this paper is structured as follows. Section 3 describes how weather data is made available, its format, and internal structure. Section 4 presents the system architecture

[1] https://trackandknowproject.eu/
[2] https://bigdatastack.eu/
[3] http://datacron-project.eu/

of the weather integration service. Section 5 provides various extensions of the basic functionality, thus improving the usability of the system in different application scenarios. The experimental evaluation is provided in Section 6, and we conclude the paper in Section 7.

2 RELATED WORK

The significance of integrating mobility data, in the form of AIS messages, with weather data has been identified as a major challenge for enabling advanced analytics in the maritime domain [1].

In the context of linking mobility data to external data sources, in order to produce semantic trajectories, one external source that has been considered is weather. For example, FAIMUSS [6] generates links between mobility data and other geographical data sources, including static areas of interest, points of interest, and also weather. The above system is designed to generate linked RDF data, using a RDF data transformation method called RDFGen [7], which imposes that the output must be expressed in RDF. However, this also poses an overhead to the application, since it dictates the use of an ontology and the representation of domain concepts. Instead, in our work, we focus on lightweight integration, which practically associates weather attributes to spatio-temporal positions. This approach is easier to use by developers, without imposing the use of RDF.

Only few works study the concept of weather data integration, focusing on real-time applications. Gooch and Chandrasekar [4] present the concept of integrating weather radar data with ground sensor data which will respond to emergent weather phenomena such as tornadoes, hailstorms, etc. The integration procedure takes place in CHORDS and a special technique is used in order to address the high dimensionality of weather radar data.

Kolokoloc [5] applies open-acess weather-climate data integration on local urban areas. Specifically, by using open-access data by meteo-services, integration of weather data is applied on locations stored in MySQL database.

3 DESCRIPTION OF WEATHER DATA

GRIB (Gridded Binary) format [4] is a standard file format for storing and transporting gridded meteorological data in binary form. The GRIB standard was designed to be self-describing, compact and portable. It is maintained by the World Meteorological Organization (WMO). All of the National Met Services (NMS) use this kind of standardization in order to store and exchange forecast data. GRIB files are provided by National Oceanic and Atmospheric Administration (NOAA), containing data from numerical weather prediction models which are computer-generated.

NOAA offers several data sets composed of GRIB files, based on one of the provided model data. Four categories of model data are available[5]; Reanalysis, Numerical Weather Prediction, Climate Prediction and Ocean Models. Model data are represented on a grid (two-dimensional space), divided into cells where each one maps a specific geographical area. Data is associated with every grid cell; weather information is provided for every geographical place being included in the grid. Model data contain also the temporal dimension inasmuch the weather conditions do not remain static accross the globe. In other words, model data are gridded data with spatio-temporal information. The offered data sets can be considered as three-dimensional cubes with weather data over a time period. In some cases, a fourth dimension is included, namely the altitude, for weather information that does not refer to the surface of the earth.

In this work, we use GRIB files based on the Global Forecast System (GFS) which is a type of Numerical Weather Prediction [6] (NWP) data model. NWP is a widely used model for weather forecasting generally, exploiting the current state of weather for making predictions. Current observations are (computer) processed, as they served as an input to mathematical models. Many attributes of the future weather state are produced as an output, such as temperature, humidity, precipitation, etc.

The GFS model[7] is composed of four distinct models; the atmosphere model, the ocean model, the land/soil model and the sea ice model. All of these models provide a complete image about the weather conditions around the globe. GFS is produced by the National Centers for Environmental Prediction (NCEP). The data set product type we use in this work is the GFS Forecasts. Also, two other GFS product types exist, the GFS Analysis and the Legacy NCEP NOAAPort GFS-AVN/GFS-MRF Analysis and Forecasts. The products come with some data sets that differentiate to the grid scale or the covering time period.

In this work, we use the data set of GFS Forecasts product that has the globe partitioned per 0.5° degrees on the geographic coordinates (longitude and latitude); also, another GFS Forecasts product exist that has the globe partitioned with 1° degrees. Every day the mathematical forecast model is run four times and has one of the following time references; 00:00, 06:00, 12:00 or 18:00. The time reference is the time that the forecast starts. Each of the forecast models cover the weather conditions around the globe for 384 hours (16 days) after its starting time. Specifically, a forecast model covers the weather conditions for 93 different timings, called steps. Every step is a distinct GRIB file, containing numerous weather attributes that are instantaneous and aggregates (averages).

The steps start from 000 to 240 (increased by 3) and continue to 252 until 384 (increased by 12). The step number indicates that the weather information contained in the GRIB file refer to the timing of X-hours after the forecast starting time. For example, the 000 step contains only instantaneous variables, referring to the forecast starting time (as the first step, it does not contain aggregate variables). The 003 step contains both instantaneous and aggregate variables. The instantaneous variables refer to the timing of weather attributes after 3-hours from the forecast reference time. The aggregate variables contain the averages of the 3-hours that passed. The same applies to the 006 step, containing both instantaneous and aggregate variables. The instantaneous variables refer to the timing of weather attributes after 6-hours from the forecast reference time. The aggregate variables contain the averages of the 6-hours that passed. The same pattern does not continue for the aggregate variables on the next steps. For instance, the 009 step contain aggregates variables that refer to the averages of weather attributes of the last 3-hours and step 012 contain aggregate variables that refer to the averages of weather attributes of the last 6-hours (the pattern is repeated until the 240 step). The aggregate variables of the steps greater than 240 [252...384] are the averages of weather attributes of the last 12-hours.

In our work, we use the four forecast models of a day with step 003 from the GFS Forecasts product; therefore, every day

[4] http://weather.mailasail.com/Franks-Weather/Grib-Files-Getting-And-Using
[5] https://www.ncdc.noaa.gov/data-access/model-data/model-datasets
[6] https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/numerical-weather-prediction
[7] https://www.ncdc.noaa.gov/data-access/model-data/model-datasets/global-forcast-system-gfs

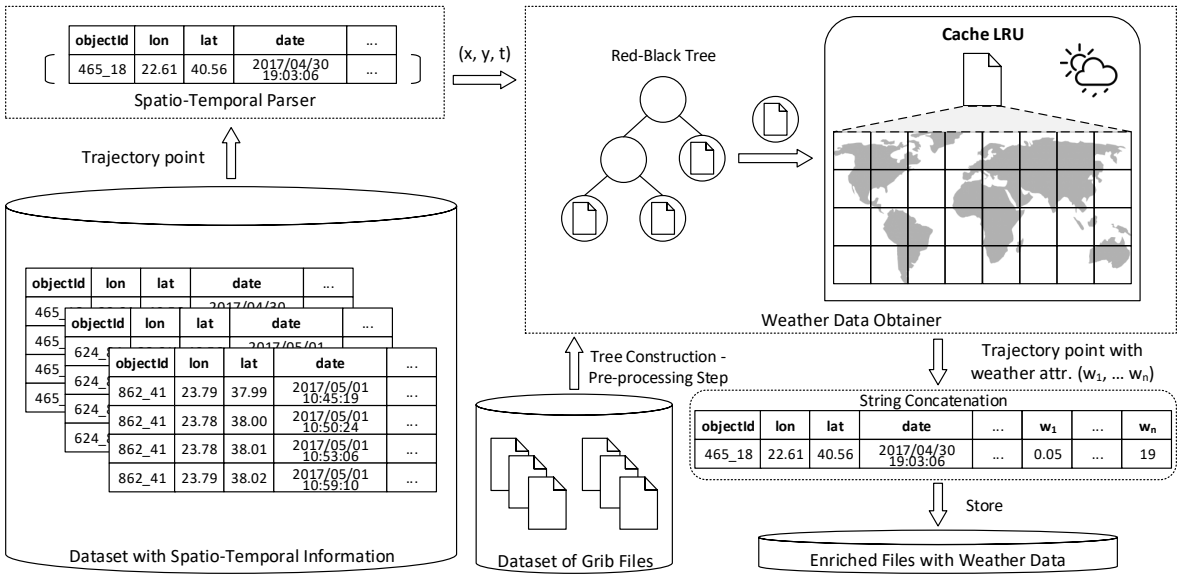

Figure 1: The system architecture of the weather integrator mechanism.

consists of 4 GRIB files that cover the instantaneous variables of weather attributes at specific timings of a day; 3:00, 9:00, 15:00 and 21:00. Since we use the 003 step, we have access to the 3-hour aggregates variables that cover the time following time periods of a day; 00:00-3:00, 06:00-9:00, 12:00-15:00 and 18:00-21:00.

4 SYSTEM ARCHITECTURE

The proposed system operates at the level of a single record, corresponding to a spatio-temporal position, and processes records independently of each other. The spatio-temporal position can be in the 2D space (x, y, t) of 3D space (x, y, z, t). At an abstract level, the system uses an external source storing weather data, in order to extract the desired weather attributes $w_1, w_2, \ldots, w_n$ that are associated with the specific position. In the case of 2D data, its output is an extended record that consists of the fields: $x, y, t, w_1, w_2, \ldots, w_n$. Obviously, any other additional fields of the input record are maintained also in the output record. In the following, we present our techniques for implementing this data integration process in an efficient way.

4.1 Basic Functionality

The architecture of the system consists of two parts-mechanisms whose functionality is combined for the data integration service provision. The first part is called Spatio-Temporal Parser and the second part is the Weather Data Obtainer. The overall architecture is illustrated in Figure 1.

The Spatio-Temporal Parser parses sequentially the records of the input data set of mobility data. For each record, a set of basic data cleaning operations are performed. For instance, the spatio-temporal part is checked both for its existence (null or empty values) and its validity (valid longitude and latitude values). If the spatial or temporal information of a record is out of the valid range or missing, the parser ignores the whole record, writes information in an error log, and the parsing procedure continues by accessing the next record. Each record with valid spatial and temporal information is passed to the Weather Data Obtainer Mechanism, which is responsible of getting the values of the weather attributes from the weather data source that contains weather information (GRIB files). Then, the obtained values are concatenated with the current processed record, forming thus an enriched record containing values of weather attributes. Subsequently, the resulting record is written to a new file in the hard disk; the whole procedure generates a new (enriched) data set. The logical separation of parsing from the remaining functionality is useful, since the system can be easily extended to read data from other data formats and sources, such as XML, JSON, or a database.

The Weather Data Obtainer is the mechanism that finds the values of weather attributes given a longitude, a latitude and a date value. In case of 4D mobility data, it also uses the altitude as input. The functionality of the Obtainer is based on GRIB files since its role is to obtain from them weather information for a specific timespan. After the Obtainer has received the spatial and temporal information, its first step is to determine the right GRIB file that should be accessed in order to get the values of the weather attributes; each of the GRIB files contains weather data only for a specific time period. As a result, the covering timespan of the chosen GRIB file should be the closest to the given timestamp of the spatio-temporal position.

The procedure of matching the given timestamp with one of the GRIB files, is achieved by maintaining a Red-Black Tree data structure in-memory, organizing the references (paths) of each GRIB file from a given set. The tree's node arrangement (key) is determined by the covering time of each GRIB file. Given a timestamp such as 4/12/2016 05:10:00, the tree finds two GRIB Files - f_1 and f_2 that cover earlier and later time respectively; in our example, these are 4/12/2016 00:00:00 (f_1) and 4/12/2016 06:00:00 (f_2). Due to the fact that the given timestamp is closer to 4/12/2016 06:00:00, the f_2 file is chosen for opening. The formation of the Tree Data Structure is considered as a pre-processing step, prior to processing mobility data.

After a specific GRIB file is selected, it must be opened in order to retrieve the weather attributes associated with the spatial part of the spatio-temporal position. There are two options of accessing the values of weather attributes of a GRIB file. The first is by loading and keeping in memory the weather attribute(s) of interest, while the second is by retrieving the weather attribute(s) from disk. The purpose of loading and keeping in memory is to perform efficiently repeated read operations, but there is a natural trade-off in terms of speed and main memory consumption. In our case, the parameters required to identify the value of a weather attributes are the spatial values (longitude and latitude) of the record at hand. These are used for determining the cell (region that results from grid partitioning) in order to get the values of the weather attributes.

4.2 Caching Mechanism

As GRIB files are binary files, which are accessed by an API, there exists an overhead related to opening a file. In case of multiple read operations, this cost can easily dominate the total processing time, especially if many files need to be opened by input records.

To avoid this overhead, we introduce a simple caching mechanism, which practically maintains in memory references to open files, in order to avoid repeated open/close operations. In the general case, the caching mechanism is crucial for the performance of the data integration procedure because a GRIB file remains open and can serve many sequential requests. This relieves the Weather Data Obtainer from the task of re-loading the GRIB file in-memory for every record, thus saving significant time. The cache replacement policy adopted is simple LRU.

It should be mentioned that in case of sorted access to mobility data based on the temporal part of the spatio-temporal position, the value of the caching mechanism is negligible. However, there exist cases where the underlying mobility data is not strictly sorted by time. This typically occurs in real-life surveillance data acquisition, where some records corresponding to positions of moving objects may be delayed. In such cases, the caching mechanism can improve performance significantly.

5 EXTENSIONS

In this section, we describe two extensions of the basic system for weather data integration. The first extension concerns integration of weather information with complex geometries, such as 3D sectors and airblocks, which are prevalent in air-traffic management (ATM) applications. The second extension is about providing output in RDF format, thereby generating linked data.

5.1 Enriching Complex Geometries with Weather Data

A useful extension of the proposed system is towards more complex (compared to a single point) geometries. In many cases we may need to associate the trajectory of a moving object (i.e. a LineString geometry) with weather conditions, or a region or a cluster of regions on the surface of the earth (i.e. polygon or multi-polygon geometries).

The first extension of the proposed system is towards computing the average of values of a selected weather attribute, over a (potentially 3D) geometry. Specifically, given a geometry g and a time interval $[t_s, t_e]$ (not necessarily a proper interval, i.e., it may hold that $t_s = t_e$), where t_s is the time instant denoting the starting time of the interval and t_e is the time instant that the interval ends, this extension returns the average of all the values retrieved for all the points of the geometry. Since the GRIB file that we use has resolution of 0.5 degrees, we reduce the geometry to 0.5 degree precision. This will reduce the number of points and the number of requests to the GRIB file. The same geometry simplification is applied for altitude of the 3D geometry, i.e., the z-axis values are reduced to the isobaric levels used in the GRIB file (and for weather attributes that depend on altitude). The core function used for retrieving the values of selected weather attributes for a given spatio-temporal position is used for each point of the geometry, and the average of these values is returned as result.

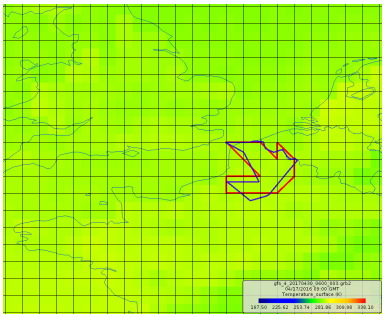

Figure 2: Example of airblock (in blue), its simplified geometry (in red), and temperature surface shown in the coloured map.

Figure 2 depicts the 2D projection of a selected 3D geometry (corresponding to an airblock) located above Lille, in France. Computing the value of a selected weather attribute (e.g., temperature) for this airblock is performed by taking the points that constitute the simplified geometry, and retrieving the weather attribute values for these points. As depicted in the figure, the geometry spans multiple cells of the GRIB file with different temperature values (shown in the coloured map), thus different temperature values are retrieved, and the average is computed.

A generalization of this extension, is to return a vector of values for each selected weather attribute. This feature is useful especially for LineString geometries, i.e., for studying the behavior of a moving object through its recorded trajectory. Then, the application can specify how to use the vector to compute the weather conditions. Averaging the values of the vector is just the most straightforward use.

5.2 Providing Linked Data in RDF

The proposed system has also been extended to operate on RDF data, both as a consumer and as a server. Since RDF is the W3C standard to be used for Linked Open Data, connecting as a consumer to RDF triples, it can exploit any positioning data available on the web, to return it enriched with weather data. Furthermore, serving RDF positioning data enriched with weather data under a common schema, can support several tasks from event or pattern recognition to link discovery between multiple data sets. Exactly this functionality of our system to generate linked mobility data with weather has been exploited in the context of the datAcron project [6, 8].

When acting as an RDF consumer and given the schema of the data source, the proposed system executes a SPARQL query to retrieve the necessary positioning and temporal values for each point. If the data source provides complex geometries, these can also be exploited, as discussed previously. For each record

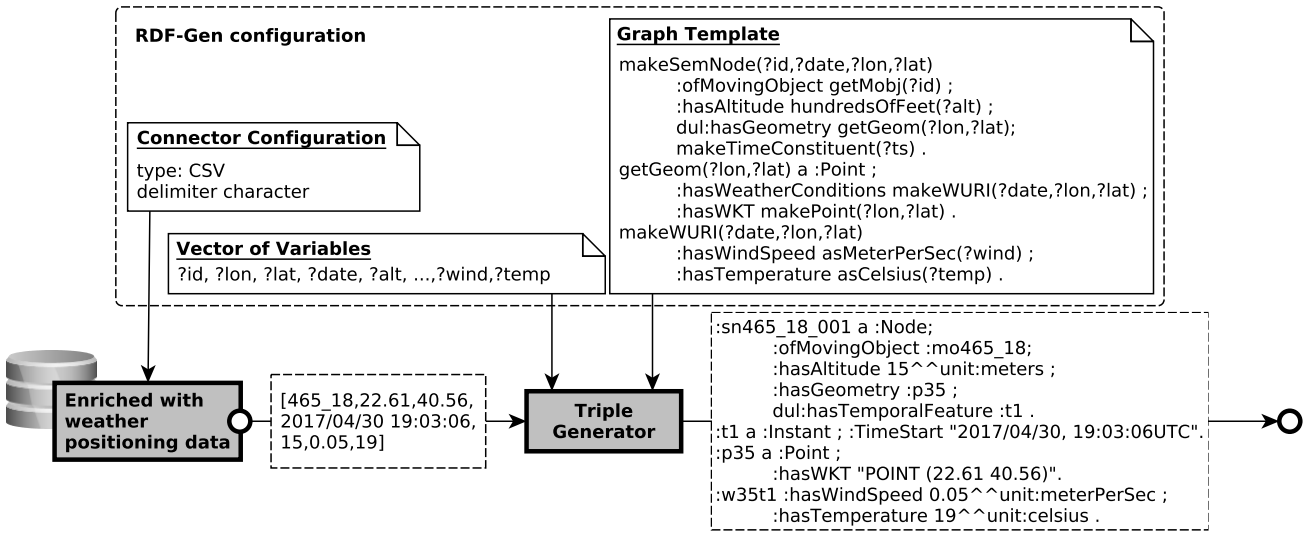

Figure 3: Example of generating linked RDF data.

retrieved by the SPARQL query, the core function retrieves the corresponding values of the selected weather attributes and enriches the record.

When the proposed system serves the enriched data as RDF triples, it only requires a triple template to be used on an RDF-Gen [7] instance. This will transform each geometry that is enriched with weather data, to RDF triples w.r.t. a given schema. Figure 3 illustrates the operation of our system as a server of RDF data. The records describing positional information have been extended with the desired weather attributes. The triple generator of an RDF-Gen instance receives such records, and outputs a corresponding RDF graph fragment, which has been specified by a graph template. Essentially, the graph template determines the structure of the output RDF data. In addition, it supports data transformation functions, such as *makeSemNode* in the graph template depicted in Figure 3.

Obviously, the proposed system can be used in the same time as consumer and as a server, enriching RDF data with weather attributes and values. Since RDF-Gen can provide consistent triples to any given schema, this extension can enhance any positioning and weather related ontology.

6 EXPERIMENTAL EVALUATION

In this section, we provide the results of the empirical evaluation performed using real-life data from the urban domain, provided by a fleet management data provider. All experiments were conducted on a computer equipped with 3.6GHZ Intel core i7-4790 processor, 16GB DDR3 1600MHz RAM, 1TB hard disk drive and Ubuntu 18.04.1 LTS operating system. Our code is developed in Java and is available at the following link: https://github.com/nkoutroumanis/Weather-Integrator. For the access to GRIB files, we use the NetCDF-Java Library[8].

6.1 Experimental Setup

Data sets. The mobility data set used in this work for the application of the data integration procedure is in the form of CSV files, containing real trajectories of vehicles in the region of Greece.

[8]https://www.unidata.ucar.edu/software/thredds/current/netcdf-java/documentation.htm

Figure 4: Illustration of mobility data use in the experimental evaluation.

Figure 4 provides an illustration of the data distribution on the geographical map. Each record constitutes a spatio-temporal point with additional information of the vehicle, such as speed, fuel level, fuel consumed, angle, etc. The records of the CSV files are provided in temporal sort order. This resembles real-life operation, since positions of moving objects are transmitted by devices located on vehicles, even though they are not received in strict temporal order, but with small discrepancies.

Due to the fact that the size of the complete data set is about 130GB and spans one year (July 2017 - June 2018), we take a small sample (consisting of 9 CSV files) whose temporal part is in the time period of January 2018 for performing the first set of experiments. Each file is approximately 550KB and contains about 4,500 records. In addition, we use a larger sample that consists of 10GB of data, having 81,483,834 records in total.

For the weather data, we downloaded 124 GRIB files (31 x 4) corresponding to January 2018. These files were used for obtaining weather data information in the data integration process. We use only the 003 steps of the 4 forecast models per day. The total size of the GRIB files is 8GB.

The resultant (enriched) data set contains the following 13 weather attributes that describe promptly the rain-related weather conditions:

- Per cent frozen precipitation surface
- Precipitable water entire atmosphere single layer
- Precipitation rate surface 3 Hour Average
- Storm relative helicity height above ground layer
- Total precipitation surface 3 Hour Accumulation
- Categorical Rain surface 3 Hour Average
- Categorical Freezing Rain surface 3 Hour Average
- Categorical Ice Pellets surface 3 Hour Average
- Categorical Snow surface 3 Hour Average
- Convective Precipitation Rate surface 3 Hour Average
- Convective precipitation surface 3 Hour Accumulation
- U-Component Storm Motion height above ground layer
- V-Component Storm Motion height above ground layer

Metrics. Our primary target is to make the integration procedure efficient. For this purpose, we use the following metrics that reflect the mechanism performance:

- Execution time: The total required time for the completion of the integration procedure (in minutes).
- Throughput: The number of processed records per second (rows/sec).
- Cache hit ratio (CHR): The ratio number of cache hits to the total number of records. In other words, this number is the percentage of records that have been enriched with weather information without requiring the corresponding GRIB file to be loaded in-memory. The higher the CHR value, the larger the benefit in execution time.

Methodology. The experimental evaluation is structured as follows. First, we evaluate the performance of our system, in terms execution time and throughput. Thus, we use two samples of the complete data set of different size:

- *Small data set*: 5.1MB of data corresponding to the trajectories of few vehicles in January 2018.
- *Large data set*: 10GB of data covering the time span of the complete data set.

The size of the integrated data set with weather is 8.6MB and 16.6GB respectively for the two sample data sets above.

Second, we evaluate the performance of the caching mechanism. Since data is provided sorted in time, the caching mechanism is of little use. Therefore, we randomly shuffle the input records, thus making a worst-case scenario where the input data are processed in random order. In this case, two consequent records will access different GRIB files with high probability.

Table 1: Evaluation on small data set.

Weather Integration	Execution Time	Memory Consumption	Throughtput
With Indexing	12 sec	229 MB	3,570 rows/sec
Without Indexing	1,660 sec	106 MB	26 rows/sec
Pre-processing	1 sec	57 MB	N/A

Table 2: Evaluation on large data set.

Procedure	Execution Time	Memory Consumption	Throughtput
Weather Integration	29,261 sec	176 MB	2,784 rows/sec
Pre-Processing	5 sec	59 MB	N/A

6.2 Evaluation of Basic Functionality

Table 1 demonstrates some elements about the data integration procedure for the case of the small data set. The first row in the table refers to keeping in-memory the retrieved weather values, whereas the second row corresponds to retrieval from disk. Clearly, the former is the most efficient way to perform the integration task, achieving throughput of 3,570 rows/sec. Instead, the latter approach only processes 26 rows/sec. This gain comes with an overhead in memory consumption, which is almost doubled, but is still manageable. Notice that the input data set is provided sorted in time, therefore the observed throughput of 3570 rows/sec is the best performance that can be achieved on the given hardware. Regarding the pre-processing overhead, namely the construction of the Red Black Tree that indexes the GRIB files, this is in general negligible (see the third row of the table).

Table 2 shows the results when the large data set (10GB) is used. We only employ our approach with in-memory maintenance of the retrieved weather values. Again, the throughput is quite high (2,784 rows/sec), thus showing that our performance results also hold in the case of large data sets.

6.3 Evaluation of Caching Mechanism

Figures 5, 6, 7 and 8 show the performance achieved when the caching mechanism is put in action. To stress-test our system, we randomly shuffle the input records, thus generating a mobility data set in random temporal order. Notice that is the hardest setup for our system, since two consequent records will need to access different GRIB files with high probability. To evaluate the performance of the cache we gradually increase the cache size (depicted on the x-axis).

Figure 5 shows that we achieve a cache hit ratio of more than 80% when using a cache size of 60 entries. Much smaller cache sizes lead to lower cache hit ratio, which also has an impact on performance. Also, the total execution time is reduced by a factor of 5 when using the larger cache size, as shown in Figure 6. Figure 7 shows the achieved throughput of approximately 140 rows/sec with cache size of 60. We note that this value corresponds to the worst-case setup for the weather integration mechanism, which

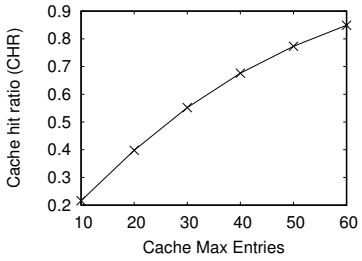

Figure 5: Cache hit ratio for increased cache size.

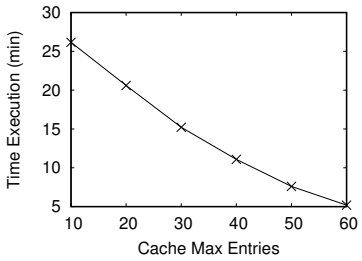

Figure 6: Total execution time for weather integration for increased cache size.

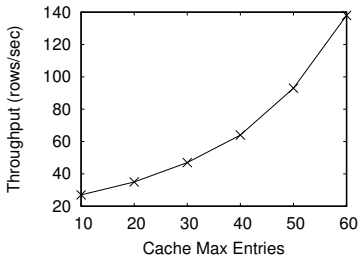

Figure 7: Throughput for increased cache size.

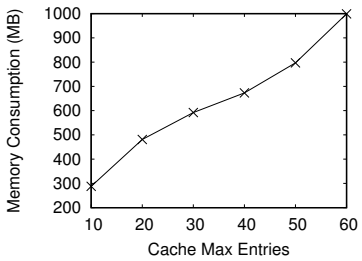

Figure 8: Memory consumption for increased cache size.

is seldom encountered in practice. Finally, Figure 8 shows that higher cache sized also result in higher memory consumption.

In summary, the caching mechanism can be very useful in the case of input data that is not temporally sorted, since it improves performance significantly, at the expense of higher memory consumption.

7 CONCLUSIONS

In this paper, we presented a system for integrating mobility data with weather information, which focuses on ease of use and efficiency. The proposed mode of operation is record-by-record, which is an abstraction that offers significant benefits, including fairly easy parallelization. Furthermore, we show that the proposed system is extensible, demonstrating its use to enrich complex 3D geometries with weather (instead of simple points) and the generation of linked data in RDF. Our experiments on real-life data sets show the efficiency of our system.

In our future work, we intend to study the gain in performance that can be attained by means of parallel processing, using a Big data framework, such as Apache Flink of Spark Streaming. Moreover, we will focus on different use-cases where our system can be applied, e.g., batch processing to enrich a vast database of historical trajectories with weather information. Also, we will explore in much more detail the issue of enriching complex geometries with weather information, which is not straightforward even for domain experts, especially for large-sized objects that cover large parts of the space (trajectories, sectors, etc.).

ACKNOWLEDGMENTS

This work is supported by projects datAcron, Track&Know, BigDataStack, and MASTER (Marie Sklowdoska-Curie), which have received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 687591, No 780754, No 779747 and No 777695 respectively.

REFERENCES

[1] Ernest Batty. 2017. Data Analytics Enables Advanced AIS Applications. In *Mobility Analytics for Spatio-Temporal and Social Data - First International Workshop, MATES 2017, Munich, Germany, September 1, 2017, Revised Selected Papers*. 22–35.

[2] Xin Luna Dong and Divesh Srivastava. 2015. *Big Data Integration*. Morgan & Claypool Publishers.

[3] Christos Doulkeridis, Nikos Pelekis, Yannis Theodoridis, and George A. Vouros. 2017. Big Data Management and Analytics for Mobility Forecasting in datAcron. In *Proceedings of the Workshops of the EDBT/ICDT 2017 Joint Conference (EDBT/ICDT 2017), Venice, Italy, March 21-24, 2017.*

[4] Ryan Gooch and Venkatachalam Chandrasekar. 2017. Integration of real-time weather radar data and Internet of Things with cloud-hosted real-time data services for the geosciences (CHORDS). In *2017 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2017, Fort Worth, TX, USA, July 23-28, 2017*. 4519–4521.

[5] Yury V. Kolokolov, Anna V. Monovskaya, Vadim Volkov, and Alexey Frolov. 2017. Intelligent integration of open-access weather-climate data on local urban areas. In *9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2017, Bucharest, Romania, September 21-23, 2017*. 465–470.

[6] Georgios M. Santipantakis, Apostolos Glenis, Nikolaos Kalaitzian, Akrivi Vlachou, Christos Doulkeridis, and George A. Vouros. 2018. FAIMUSS: Flexible Data Transformation to RDF from Multiple Streaming Sources. In *Proceedings of the 21th International Conference on Extending Database Technology, EDBT 2018, Vienna, Austria, March 26-29, 2018*. 662–665.

[7] Georgios M. Santipantakis, Konstantinos I. Kotis, George A. Vouros, and Christos Doulkeridis. 2018. RDF-Gen: Generating RDF from Streaming and Archival Data. In *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics, WIMS 2018, Novi Sad, Serbia, June 25-27, 2018*. 28:1–28:10.

[8] George A. Vouros, Akrivi Vlachou, Georgios M. Santipantakis, Christos Doulkeridis, Nikos Pelekis, Harris V. Georgiou, Yannis Theodoridis, Kostas Patroumpas, Elias Alevizos, Alexander Artikis, Christophe Claramunt, Cyril Ray, David Scarlatti, Georg Fuchs, Gennady L. Andrienko, Natalia V. Andrienko, Michael Mock, Elena Camossi, Anne-Laure Jousselme, and Jose Manuel Cordero Garcia. 2018. Big Data Analytics for Time Critical Mobility Forecasting: Recent Progress and Research Challenges. In *Proceedings of the 21th International Conference on Extending Database Technology, EDBT 2018, Vienna, Austria, March 26-29, 2018*. 612–623.