ADMSv2: A Modern Architecture for Transportation Data Management and Analysis

Chrysovalantis Anastasiou, Jianfa Lin, Chaoyang He, Yao-Yi Chiang, Cyrus Shahabi
Integrated Media Systems Center, USC Viterbi School of Engineering
Los Angeles, California, USA
{canastas,jianfali,chaoyang.he,yaoyic,shahabi}@usc.edu

ABSTRACT

This paper presents ADMSv2, an end-to-end data-driven system that enables real-time and historical data analytics and machine learning tasks over big, streaming, spatiotemporal data. ADMSv2 employs a unified multi-layered architecture that integrates several open-source frameworks to collect, store, manage, and analyze a variety of data sources, including massive traffic sensor data, bus trajectory data, transportation network data, and traffic incidents data. ADMSv2 enables numerous applications in intelligent transportation, urban planning, public policy, and emergency response, all of which are critical for city resilience. Here, we demonstrate three application scenarios running on top of ADMSv2 to showcase the efficiency of its capabilities of query processing on real-world streaming and historical data as well as real-time data analysis using deep learning for traffic forecasting.

CCS CONCEPTS

• **Information systems → Online analytical processing engines**; **Stream management**; **Data warehouses**.

KEYWORDS

datasets, transportation, analytics, neural networks, smart city

ACM Reference Format:
Chrysovalantis Anastasiou, Jianfa Lin, Chaoyang He, Yao-Yi Chiang, Cyrus Shahabi. 2019. ADMSv2: A Modern Architecture for Transportation Data Management and Analysis. In *2nd ACM SIGSPATIAL International Workshop on Advances on Resilient and Intelligent Cities (ARIC'19), November 5–8, 2019, Chicago, IL, USA*. ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3356395.3365544

1 INTRODUCTION

The recent advances in wireless technologies, as well as the widespread usage of sensors, have led to a massive amount of real-time generated data. In many metropolitan cities, traffic sensors such as loop detectors are deployed on both highway and arterial roads to collect traffic information, such as traffic volume and speed. These sensors enable a variety of Intelligent Transportation System (ITS) for effective monitoring, decision-making, and management of the transportation network. However, the large volume, variety, and velocity of these data make the integration, visualization, and analysis an intrinsically challenging problem [5, 9].

A pioneer system for managing transportation data is PeMS [2], short for Performance Evaluation Monitoring System, which collects and archives highway traffic data in a single Relational Database Management System (RDBMS). The main goal of PeMS is to convert highway sensor data into intuitive tables and graphs that show traffic patterns as well as estimate the travel time on highway links based on historical data. Furthermore, PeMS can generate contour plots of speed versus time, which can be used by city planners to spot bottlenecks. However, PeMS only supports a predefined set of simple analytics and visualizations of transportation metrics, and its centralized architecture does not allow for scalability. Another notable system for managing heterogeneous urban data is VaVeL [1]. The VaVeL system is based on open-source big data frameworks, but the system is not available to the public.

Previously, our research center has developed TransDec-ADMS [3] (short for Transportation Decision-Making Archived Data Management System), a system that can support spatiotemporal queries through an interactive web-based interface over a large transportation database. ADMS is backed by the Oracle Spatial RDBMS [8], which can efficiently process a variety of real-time and historical spatiotemporal queries by leveraging the state-of-the-art index structures implemented in the RDBMS. However, as the size of the data increases, ADMS incurs longer response times. Furthermore, its centralized architecture cannot scale to support a variety of applications that utilize the archived historical data of large volumes or real-time data with high frequency.

PeMS and ADMS were both developed in the pre-cloud era and hence do not utilize the state-of-the-art data management and analysis systems and tools available now. Furthermore, they were both designed to operate on top of centralized database systems and, inherently, cannot scale well to accommodate the increasing size of datasets. To address these challenges, this paper presents ADMSv2 (version 2), which is a multi-layered end-to-end system that enables both historical and real-time analytics and machine learning tasks over big, streaming spatiotemporal data. Unlike its predecessor, ADMSv2 adopts an entirely new architecture that embraces open-source big data frameworks. The ADMSv2 multi-layered architecture allows for the independent development, deployment, and maintenance of each layer. Furthermore, the architecture allows for better separation of concerns, i.e., each layer is responsible for distinct functionalities and features of the system.

In the design and development of ADMSv2, we have two goals in mind. First, for all queries, whether continuous queries on streams, historical queries, or complex analytics, the system needs to provide

ARIC'19, November 5–8, 2019, Chicago, IL, USA

ACM ISBN 978-1-4503-6954-1/19/11…$15.00
https://doi.org/10.1145/3356395.3365544

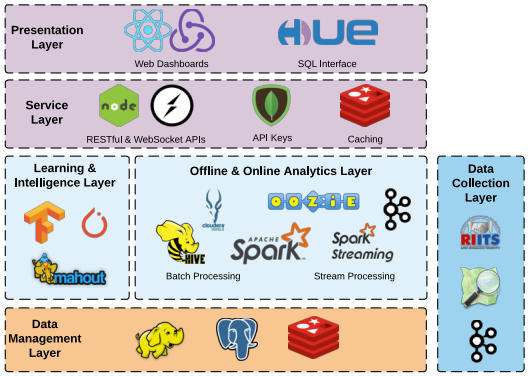

Figure 1: The ADMSv2 multi-layered architecture and the open-source frameworks in each layer

reasonable response times (~500 milliseconds for real-time data, and ~ several seconds for analytical queries on large historical data). Second, as the size of the datasets increases continuously, the system needs to maintain these low response times. ADMSv2 achieves both these goals and, hence, supports a new set of interactive data-exploration applications that would not be possible in the previous version. We showcase the potential of our system by demonstrating several applications with ADMSv2 that show the importance and capabilities of each layer of the system. Specifically, we develop dashboards for historical queries, a traffic forecasting interface for analysis and machine learning, and a map-based dashboard for real-time bus tracking and estimated arrival time for continuous queries over streaming data.

Our partnership with the Los Angeles Metropolitan Transportation Authority (LA Metro) enables ADMSv2 to access large and high-resolution (both spatial and temporal) traffic sensor data from a number of transportation authorities in Southern California. This dataset includes both sensor metadata and real-time data for freeway and arterial traffic sensors (~16,000 loop-detectors) covering 4,300 miles, 2,000 bus and train automatic vehicle locations (AVL), and ramp meters with an update rate as high as 30 seconds. The dataset also includes data about several incident types (such as accidents, traffic hazards, and road closures) reported at a rate of approximately 400 incidents per day. We have been continuously collecting and archiving the datasets mentioned above for the past eight years, and with an annual growth of 1.5TB, ADMSv2 is the most massive traffic sensor data warehouse in Southern California. Through our dashboards, city officials are able monitor traffic and incidents in real-time and use predictive traffic analysis which are important for city resilience and more informative decision-making such as preemptive planning during unexpected major incidents.

2 ADMSV2 ARCHITECTURE

This section presents the ADMSv2 architecture and discusses the challenges that ADMSv2 aims to tackle by adopting modern open-source frameworks.

2.1 Overview

ADMSv2 employs a multi-layered architecture in which each layer is designed to fulfill a distinct objective. This multi-layer architecture is flexible and easy to maintain since every layer has a predefined, structured input and output format. Within each layer, we build modular components that can be easily modified as new technologies become available without rebuilding the entire system. Figure 1 depicts the ADMSv2 multi-layered architecture. The *Data Collection Layer* includes several adapter components for consuming data. The *Offline & Online Analytics Layer* contains both a scalable message queue (e.g., Apache Kafka) and a cluster computing framework (e.g., Apache Spark) to support a variety of analytic tasks. The *Learning & Intelligence Layer* enables efficient development and deployment of machine learning tools over ADMS data. The *Data Management Layer* contains specialized data storage systems for accessing 1) analytic data requiring spatial and or temporal filtering (e.g., PostgreSQL), 2) recent/real-time data requiring fast data access (e.g., Redis), and 3) historical data requiring distributed computing for data aggregation and filtering (e.g., Hadoop HDFS). The *Service Layer* includes several data access APIs for consuming analytic results in ADMS, e.g., traffic forecasting and bus arrival time estimation. The *Presentation Layer* serves web dashboards and maps for visualizing and disseminating ADMS data.

2.2 Data Collection Layer

This layer is responsible for consuming data from external data providers and delivering them to the Data Management Layer for archival. Although the task sounds trivial, there are two main challenges. First, a data provider might require that their data stream is accessed through a provided Software Development Kit (SDK) developed in a specific programming language, whereas, some other data provider might offer a freely accessible web service. Second, not all agencies generate data in the same velocity and volume and, therefore, a data collection service might struggle keeping up. ADMSv2 adopts a micro-services architecture to tackle these challenges. This means that for each data type, i.e., congestion, bus, traffic incidents, ADMSv2 employs a separate data collection service (micro-service) on the data collection layer. For each micro-service, we are free to use any programming language or SDK depending on the requirements of the data provider, hence solving the first challenge. To tackle the second challenge, ADMSv2 takes advantage of a scalable message queue system, Apache Kafka, which allows deploying a micro-service that fetches data from a data provider while a group of micro-services consumes and parses those data.

2.3 Data Management Layer

Being the base of all other layers, the Data Management Layer is a crucial layer with the purpose to archive and index the collected data while providing the means necessary for accessing them. The fact that all consumed data have both spatial and temporal dimensions along with their vast heterogeneity, massive volume, and high velocity, makes storage and retrieval a particularly challenging task. We want queries of all types, i.e., queries that involve either spatial, temporal, or both, to be efficiently handled by this layer because the response times also affect the other layers. Hence, on this layer, ADMSv2 utilizes specialized data storage systems. For spatial analysis queries, ADMSv2 deploys PostgreSQL with the PostGIS extension, an RDMS that supports spatial data and state-of-the-art spatial index structures such as R-tree [4]. For recent and real-time data that require fast data access, ADMSv2 utilizes Redis,

a main memory key-value store. For complex analytical tasks, such as join operations of sensor metadata with sensor data to perform historical aggregations and filtering, that require distributed computing, ADMSv2 employs a Hadoop HDFS cluster where we store all data partitioned on their temporal attribute. An Apache Hive metastore is also deployed on top of HDFS to provide an SQL-like interface for analyzing the archived data sets in HDFS.

2.4 Offline & Online Analytic Layer

Providing transportation insights like live traffic and historically slowest and fastest highways at different times of the day is of utmost importance not only for effective route planning but also for urban planning. Hence, ADMSv2 needs to provide support for such tasks. This layer is responsible for providing the means to process and analyze not only historical but also streaming data to generate rich insights that aid in decision-making. The challenge here is that as an analytics task becomes more complex, e.g., analyzing a massive historical bus trajectory dataset to estimate the time of arrival of a bus to the next station and how traffic affects the delays, the response time and the amount of resources required increases. Therefore, on this layer, ADMSv2 deploys Apache Spark, an in-memory distributed processing engine, which enables fast analytics on top of a distributed HDFS cluster. ADMSv2 also leverages Spark Streaming to consume and process streaming data from Apache Kafka. The combinations of these technologies enables ADMSv2 to realize many analytical functionalities, e.g., producing average speed of each highway sensor in the past five(or more) minutes, finding time-dependent top five most (and least) congested highway segments, buses with large schedule deviation, in a matter of seconds in the worst case even though thousands, or millions, of data records need to be evaluated.

2.5 Learning & Intelligence Layer

The recent advances in machine learning in combination with our massive datasets, allow us to train and deploy high-quality prediction and forecasting models. One of the inherent challenges of all learning tasks is the preparation and cleaning of datasets and the efficient extraction of features needed for learning. Moreover, the training phase requires careful implementation of learning algorithms to achieve convergence without wasting resources. For the former challenge, ADMSv2 utilizes the data access methods of the Data Management layer to prepare our data sets for learning, and stores and shares prepared data sets between different learning tasks. For the latter, ADMSv2 deploys well-known and established learning frameworks such as TensorFlow and PyTorch on this layer to allow for the quick development of machine learning models.

2.6 Service & Presentation Layers

All ADMSv2 layers are kept internally and considered as the backend. Therefore, to provide public access to the rich datasets and machine learning models, we need a layer that puts everything together without requiring the end-user to understand the complexity of the system. The Service Layer serves as a gateway to the data warehouse by means of RESTful APIs. Through these APIs, one can access aggregated historical data and analytics, real-time data, and predictions from forecasting models. For the dissemination of all the insights from the data warehouse, a set of web dashboards is deployed on the Presentation Layer. These dashboards include web maps for the visualization of real-time and forecasted traffic as well as real-time information about traffic incidents.

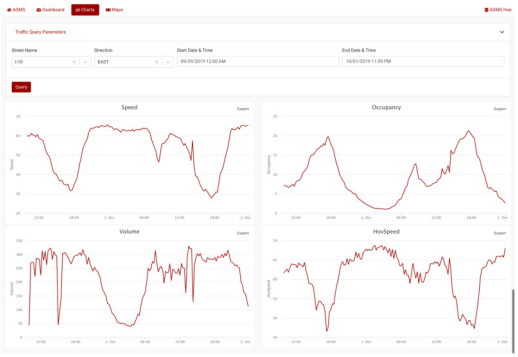

Figure 2: Traffic charts (speed, volume, occupancy, and HOV speed) for I-10 East from Sept. 29th to Oct. 1st, 2019

3 DEMONSTRATION SCENARIOS

This section presents three applications that we developed on top of the ADMSv2 architecture, i.e., traffic dashboards for both real-time and historical data, traffic forecasting, and bus estimated time of arrival. We plan to demonstrate these web dashboards at the conference to showcase the capabilities of ADMSv2.

3.1 Traffic Dashboards

The first application is a set of web-based dashboards that provide a user-friendly query interface and is designed to test the efficiency of historical aggregations on the Data Management Layer. Figure 2 is a screenshot of the Traffic Charts Dashboard. A query input section allows for selecting the desired query parameters, i.e., the name and direction of the highway (or arterial roads), the query start date and time, and query end date and time. After the query is executed on the ADMSv2 Data Management Layer, four charts are generated and displayed on the dashboard. The screenshot in Figure 2 shows the charts for speed, occupancy, volume, and HOV (high occupancy vehicles) lane speed for the days of September 29th to October 1st, 2019 on the East-bound I-10 freeway. The metrics are aggregated over all the sensors on the highway (or arterial), but one can still observe the rush-hour time window.

Our set of dashboards also includes a real-time map-based user interface that depicts the current state of the traffic and is designed to test the fast retrieval of streaming data. Figure 3 displays the traffic sensors' locations color-coded based on the current state of congestion, i.e., a sensor is marked green if there is light or no traffic and dark red if it is heavy. Such a real-time interactive map is only possible because of the continuous consumption of external data streams and the caching capabilities provided by the Data Management Layer.

3.2 Traffic Forecasting

Traffic can be a nightmare, especially to the people who commute to work daily. Therefore, we take advantage of our large traffic dataset

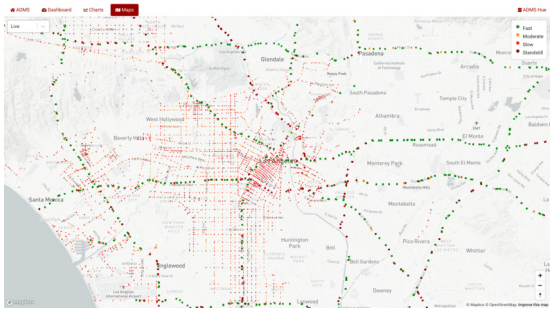

Figure 3: Real-time highway congestion map for the Los Angeles County. The colored points depict the traffic state of sensors (green for light traffic, dark red for heavy traffic)

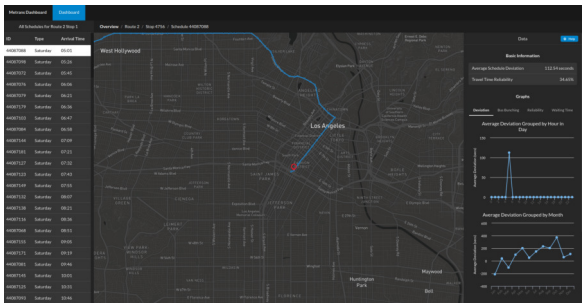

Figure 4: Dashboard for information on bus routes and estimated time of arrival of buses at stations

to train and serve a real-time traffic forecasting model. To achieve this in ADMSv2, first, we leverage the distributed storage in the Data Management Layer to efficiently clean and transform the historical sensor data to prepare a training dataset. Then, we feed this training dataset as input to the Learning & Intelligence Layer where we train our advanced deep neural network model, DCRNN[6]. The trained model is highly accurate and produces forecasts based not only on the current traffic time-series but also on real-time traffic incidents. To enable continuous forecasting, we leverage Apache Kafka. Real-time traffic updates are being consumed from the message queue and passed into the model to predict the future state of the traffic for the next 1 hour. We develop a map-based web dashboard, similar to the one in Figure 3, to display the traffic forecasting results. After clicking on one of the sensors, a window that displays the speed forecast chart appears. As time passes and real data are collected, the chart is updated to display the true speed values and the forecasting error. City officials can leverage this dashboard to better understand how real-time incidents are going to affect the traffic and allocate resources appropriately.

3.3 Bus Arrival Time Estimation

Having an accurate prediction on when the next bus will arrive can help improve the public transportation systems. Our rich bus trajectory datasets can help us model how buses deviate from the original schedule over time. For each route, we are able to estimate the delay of buses at each stop, at different times of the day. Hence, we create and deploy an analytical Spark task on the Analytics Layer that estimates the schedule deviation of buses in real-time by consuming streaming data from Apache Kafka and applying the approach described in [7]. We utilize the calculated schedule deviation estimations to build a web dashboard that provides insights about the arrival time of a bus at a specific bus stop. Furthermore, our dashboard displays the current location of buses. Figure 4 shows the front-end dashboard we develop where users can view different routes, vehicle, and station arrival time estimates.

4 CONCLUSION & FUTURE WORK

In this paper, we presented ADMSv2, an end-to-end data-driven architecture that enables real-time and historical interactive analytics and machine learning over big spatiotemporal data. The modular design of our architecture and the adoption of modern frameworks allows for more flexibility and scalability in the system. Also, we demonstrate the potential of ADMSv2 by developing several applications on top of the architecture. Although we primarily utilize transportation data for this demonstration, our system is not limited to this domain. ADMSv2 can process spatiotemporal datasets from any domain, e.g., air quality datasets, and weather datasets. As future work, we will extend the capabilities of the architecture to allow for the execution of more complex user-defined queries, such as Spark and MapReduce jobs, through a web portal providing researchers, as well as government officials, with the tools necessary for helping the community. We also plan to extend our Machine Learning and Intelligence layer to allow for caching and sharing of training features between different training tasks. In sum, ADMSv2 can be used as an academic test-bed with real-world data and applications to integrate and evaluate new spatiotemporal data management and analysis techniques.

ACKNOWLEDGMENTS

This research has been funded in part by Caltrans-65A0674, LA Metro contract LA-Safe-PS36665000, the USC Integrated Media Systems Center, and the USC METRANS Transportation Center. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of any of the sponsors.

REFERENCES

[1] 2019. VaVeL. http://www.vavel-project.eu/. [Online; Accessed 30-August-2019].
[2] Chao Chen, Karl Petty, Alexander Skabardonis, Pravin Varaiya, and Zhanfeng Jia. 2001. Freeway performance measurement system: mining loop detector data. *Transportation Research Record* 1748, 1 (2001), 96–102.
[3] U. Demiryurek, F. Banaei-Kashani, and C. Shahabi. 2010. TransDec: A spatiotemporal query processing framework for transportation systems. In *ICDE 2010*. 1197–1200. https://doi.org/10.1109/ICDE.2010.5447745
[4] Antonin Guttman. 1984. R-trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD '84*. ACM, New York, NY, USA, 47–57.
[5] HV Jagadish, Johannes Gehrke, Alexandros Labrinidis, Yannis Papakonstantinou, Jignesh M Patel, Raghu Ramakrishnan, and Cyrus Shahabi. 2014. Big data and its technical challenges. *Commun. ACM* 57, 7 (2014), 86–94.
[6] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *ICLR 2018*.
[7] Kien Nguyen, Jingyun Yang, Yijun Lin, Jianfa Lin, Yao-Yi Chiang, and Cyrus Shahabi. 2018. Los angeles metro bus data analysis using GPS trajectory and schedule data (demo paper). In *ACM SIGSPATIAL '18*. ACM, 560–563.
[8] Oracle. 2019. Oracle RDBMS. https://www.oracle.com/database/. [Online; Accessed 30-August-2019].
[9] Seref Sagiroglu and Duygu Sinanc. 2013. Big data: A review. In *2013 International Conference on Collaboration Technologies and Systems (CTS)*. IEEE, 42–47.