Real-time Sensor Data Streams

Silvia Nittel
Spatial Informatics, School of Computing and Information Science, University of Maine, USA

Abstract

Today's observation streams of large numbers of geographically distributed sensors in geosensor networks arrive continuously at powerful servers, and users are interested in analyzing the sensor data streams in real-time. This article investigates challenges in data management that arise when massive real-time data streams become available and discusses different data management technologies for managing and analyzing real-time sensor data streams.

1 Introduction

Geosensor networks [16] consist of large sets of sensor nodes deployed in geographic space. A sensor node is a combination of a computational unit and one or more configurable sensing devices. The computational unit is programmable and runs user programs that encode the user's data and analysis needs. These needs include controlling the sampling frequency and determining how to handle the sensed data, that is, how to analyze and/or store the information locally, discard it if nothing 'interesting' is detected, or send it to other nodes or the cloud.

Sensor nodes mostly use wireless communication to send observations to other nodes. One of the many variable elements of a geosensor network architecture is the communication network *topology*. Traditional geosensor networks use *mesh topologies* [3] since they are flexible, robust and scalable. In a mesh network, individual nodes are connected to each other and base stations through multihop routing. A base station is a powerful computational node that can buffer data and is often connected to the Internet. In practical cases, Internet connectivity can be costly and limited, particularly in remote areas. Therefore, multimesh networks offer a scalable, inexpensive alternative. In a multimesh network, individual nodes are connected to each other and one or more base stations via a fine-grained radio mesh network, while a second, coarser-grained mesh network connects base stations with each other and the cloud. In this way, geosensor networks can scale up to very large numbers of nodes.

Mesh topologies with data centric routing and in-network collaboration have received much attention in research. In real-world geosensor network applications, raw sensor observations are commonly streamed out of the geosensor network, and integrated and analyzed on a more powerful server outside of the network [17, 12]. This availability of potentially very large numbers of sensor nodes – streaming observations in high frequency and in real-time – has led to novel data management and data analysis challenges. Users are interested in analysis of live streams compared to historic data, dealing with 'big spatial data' effectively, and real-time sensor stream analytics. Often, the question of appropriate data management support arises, and the recent explosion of new NoSQL systems ('Not Only SQL') in the database world does not make the selection easier.

In the past, SQL-like interfaces made it convenient to express data aggregation and subsetting tasks as queries, and delegated efficient execution to the database system. Today, the 'system' behind an SQL interface can come in different flavors, such as Hadoop-based implementations for large batch jobs over massive data sets, or real-time streaming engines working alongside traditional systems based on relational technology.

In this paper, I will discuss some of the data management challenges that arise when dealing with real-time streaming of larger numbers of sensors. The remainder of this article is organized as follows: in Section 2, sensor data streams are described in more detail. Section 3 addresses general challenges that arise when dealing with real-time sensor data streams. Section 4 discusses several state of the art data management technologies such as Hadoop-GIS, spatio-temporal database systems, and data stream engines with regard to their usability for different data management problems. Finally, conclusions are drawn and recommendations are given in Section 5.

2 Real-time Sensor Data Streams

A *sensor data stream* is a time series of sensor measurements $m_{s_j} =< t_i, l_{s_j}, v_1, v_2, ...v_n >$ generated by a sensor node s_j, based on one or more of its attached sensors. To interpret a sensed value v_i correctly, we need the timestamp t_i and location l_{s_j} of the observation (and potentially, other information such as a sensor device identifier, sensor noise level, etc.). In this article, we consider time, location and sensed value as the minimum components of a sensor measurement. For practical purposes, not every sensor node is actually able to determine its own location using GPS; in this case, we assume the node is stationary and is initialized with its location during installation. The location of a sensor might also be derived indirectly from well-known locations of other individual sensors based on the sensor identifier at the server, and added to the sensor tuple on its arrival. More formally, an attribute $a_i = (n_i, D_i)$ is a pair (n_i, D_i) where n_i is the attribute name (e.g. NO_2) and D_i is a value domain for the attribute (or a data type, e.g. floating point). A spatio-temporal *relation* R_{ST} is a finite subset of the Cartesian Product of the respective domains D_i of R_{ST}'s attributes, that is $R_{ST} \subseteq D_{a_1} \times D_{a_2} \times .. \times D_{a_n}$ with the condition that at least one domain D_i is a spatial domain and a least one domain D_j is a temporal domain.

Informally, a sensor data *stream* is a continuously updating spatio-temporal relation R_{ST} with append-only tuples created by single sensor, i.e. the time series of updates of a single sensor. Alternatively, a sensor data stream can be seen as the continuously updating, spatio-temporal relation with append-only tuples of *all* sensors s_i in a geosensor network G. This means that all sensors have the same schema, i.e. list of attributes. In the context of geosensor networks, viewing a sensor data stream as all updates of geographically and thematically related sensors is powerful, since we can reason about spatio-temporal events represented by the stream of observations. Thus, a *sensor data stream* S_s is a possibly infinite multiset S_s of time-stamped spatial tuples (t_i, s), where the spatial tuples belongs to the schema of S_s and $t_i \in T$ indicates that the spatial tuple s arrived at t_i.

3 Challenges

3.1 So, what is really new about sensor data streams?

One can ask the question "So, what is really new about sensor data *streams*?" Representing the updates of an individual sensor as a time series of data records has long been common practice in geographic and scientific applications. A traditional time series of spatial data tuples and a sensor data stream are similar if we look at the data structure: both consist of collections of spatio-temporal records, but spatio-temporal relations are *finite* sets of records while sensor data streams are possibly *infinite* multisets (there may be duplicates). The difference between relations and streams is mostly due to the data delivery and processing techniques. Because of inexpensive platforms, the *geographic density* of deployed sensors is much higher today [17, 12]. Secondly, sensor nodes use wireless communication so that observations are *available* for analysis in *near real-time*. Third, the *sensing frequency* per sensor has also drastically increased: samples every few seconds are common. For applications such as volcano monitoring, the frequency can increase to several thousand measurements per second. Characterizing sensor streams as a collection of spatially dense, high-frequency sensor observations in

near real-time nevertheless still leaves room for interpretation. What does "spatially dense" and "high frequency" mean? In today's pilot applications, we find deployments of sensors with numbers varying between 200 sensors [15] to 20,000 sensors for a city-wide deployment [17], and sample rates vary between every 2 seconds to every 10 minutes. These numbers are bound to increase significantly over the next 5-10 years, with large sensor network applications becoming more pervasive. One can expect that these advances in technology will enable and inspire scientists to rethink how best to measure and sample phenomena that they are interested in, in order to collect more data and reduce uncertainty.

3.2 User Requirements

Using spatially and temporally fine-grained geosensor networks with real-time sensor streams raises several novel, interdisciplinary questions in different scientific and engineering areas such as electrical and chemical engineering, computer science, spatial information science and environmental sciences:

- **Geosensor network design**: Geosensor network design will become a challenging question as sensors become cheaper and more abundant. In this case, geosensor network design is driven by the questions of how many sensors are optimal, and at which precise locations to deploy the sensors to get the best representation of the phenomenon and/or events of interest? Other considerations include which sensor types to use, and analysis of the tradeoffs between accuracy, compatibility, price, and energy usage.

- **Real-time Sensor Stream Management**: Once a geosensor network is designed, built, programmed, deployed, and tested the question becomes how to deal with the live-streamed data? One could store the streams as files, assuming sufficient disk storage is available, and analyze the data later. However, data management is not quite that simple if a user considers real-time data analysis needs. Choosing an adequate data management tool will simplify the task of efficient and convenient real-time data analysis.

- **Spatio-temporal Data Analysis**: Typically, analyzing sensor data streams is a user's primary interest. Existing statistical methods remain unchanged with increasing data samples, however, they might require new algorithms and implementations to handle much larger data sets and to deliver real-time results. Due to the nature of the sensor web, geosensor networks can be composed of streams from different devices, often under non-standard circumstances; this fact introduces varying accuracy and bias which needs to be accounted for during analysis. Additionally, the focus of analysis will most likely to shift to spatio-temporal analysis instead of mostly spatial analysis over snapshots of spatial data. Geovisual analytics will play a more important role in analysis [6].

3.3 Data Management Challenges

With the availability of real-time sensor data streams, today's analysis methods of spatio-temporal data are not necessarily directly affected. The onslaught of sensor data streams, however, will contribute to the "big spatial data" problem, but real-time analysis itself might not be necessary for many applications. For example, a weather forecasting model will take recent data into account, and real-time forecast changes are not critical. On the other hand, a public transportation monitoring system needs real-time analysis and response (e.g. identifying accidents, traffic jams and rerouting etc.).

Novel challenges do arise with real-time analysis of sensor data streams. This type of analysis is performed over a 'window' of the most recent data stream, and data analysis has to keep up with newly arriving data. Analysis on raw sensor data streams includes looking for patterns in the raw data, cross-correlating raw streams with other sensor streams, historic data and/or model predictions, and aggregating and summarizing raw sensor data. Ideally, data management systems can provide convenient support for these tasks. We derive three main requirements towards data management support for real-time sensor data streams.

- **Heavy lifting for raw sensor streams analysis**: A data management system should be capable of providing efficient support for real-time queries over very large amounts of sensor data streams and be able to keep up with incoming data.
- **Convenient integration with traditional data**: Since understanding, analyzing and interpreting current data is often performed through correlation with historic and/or model data, data management tools should provide seamless data representation and query capabilities between real-time sensor streams and historic and model data.
- **Data model support for continuity of time**: With frequently updating sensor data streams, data model support and query languages for the dimension of *continuous time* as well as spatio-temporal concepts become more critical. Both time as well as spatio-temporal concepts require formal foundations.

4 Data Management Support for Real-time Sensor Streams

In the following, we investigate the suitability of current data management technology for supporting analysis of real-time sensor data streams. We introduce categories of current data management approaches, and discuss data characteristics as well as suitable data management options.

4.1 Hadoop-GIS

Today, much analysis of spatial data is performed using commercial or open source geographic information systems (GIS) tools such as R or Matlab. These tools are rich in libraries for geo-statistical analysis. GIS typically use files for data storage, and the task of subsetting data in or between files is up to the user. Stream data can be added as new files, but additional support might be necessary to speed up computation for large data sets. Popular tools are often based on Hadoop [9].

Definition: The Map/Reduce paradigm was introduced with Google's technology to provide fast, scalable batch processing for tremendously large data sets like the index behind the Google search engine. The Map/Reduce technique allows seamless distribution and parallelization of code execution and data partitioning across many machines or the cloud. Open source implementations of Map/Reduce are e.g. Hadoop. In recent years, some GIS functionality has been rewritten using the Map/Reduce paradigm [9] using Hadoop-like systems for batch job processing. Thus, computing very large spatial data sets is sped up significantly [1, 2, 10, 5].

Data characteristics: Hadoop/GIS tools are useful for spatial analysis of previously collected, stored and potentially very large data sets, using any type of time intervals and spatial regions with regard to data. Data is stored in files and simply structured. This type of system does not contain out-of-the box functionality to deal with data streams characteristics, but novel architectures that seamlessly integrate batch job processing with the processing real-time data [13] are under development.

Systems: Hadoop-GIS [2], Spark R [5], ESRI tools for Hadoop [11], GeoMesa [10].

4.2 Spatial and Spatio-Temporal Database Systems

Database systems (DBS) are convenient data management and declarative querying tools for large data sets. In this context, we define DBS as data management systems that are based on SQL and relational DBS technology (as opposed to SQL implement on top of Hadoop tools, also know as New SQL). In which scenarios are DBS appropriate to manage real-time sensor data streams?

Definition: *Spatial DBS* (s-DBS) are widely used today, both as open source and commercial systems. They have extensions (data types) to describe spatial data such as points, line string, polygons, networks and coverages and built-in spatial query support to make spatial search and simple spatial analysis convenient and

efficient. A *spatio-temporal DBS* (st-DBS) supports database concepts for both space and time information. It typically includes spatial types (as described above), temporal types (e.g. timestamp, interval, etc.) and potentially spatio-temporal types (e.g. a moving point or a moving region). Most of these s-DBS and st-DBS are built using relational databases systems. For higher dimensional data, array databases are available (e.g. SciDB).

Data characteristics: S-DBS are designed for transactional and stored data. New data records (inserts) are written to disk, and are immediately available for query and analysis. S-DBS are tuned towards very large datasets and high update rates (ca. 500 new tuples inserts per second). Therefore, s-DBS and st-DBS are promising candidates for applications with sensor data streams that do not generate more than 500 new tuples per second across all sensors. Real-time data can seamlessly be integrated with historic stored sensor data, and queried on-the-fly. However, most s-DBS or st-DBS do not support continuous queries and it is the user's responsibility to repeatedly restart a query. While s-DBS and st-DBS keep up with storing sensor data streams within certain update bounds, more complex analytical processing might not be executed in a timely manner for real-time analytics, since conventional algorithms heavily rely on disk based access and do not scale well to very large data sets [12, 19].

Systems: Oracle Spatial, IBM DB2, PostGIS, MySQL and others.

4.3 Data Stream Engines

Data stream engines (DSE) have been built as tools specifically to support high-throughput querying of real-time data streams with real-time answers. Besides financial and web analytics, sensor networks have also been a driving factor for DSE development.

Definition: A DSE is a data management system for real-time analytics of continuous data streams. Some DSE are similar to DBS, that is, they offer a data model language and query language so that information needs can be expressed using declarative queries. Other DSE require a heavy amount of programming to compose queries. In contrast to DBS, a DSE reevaluates a query repeatedly over newly arriving data instead of just once as in a DBS. Since most DSE are data-driven, a continuous query produces new results as long as new data arrives at the system. DSE do not support transactions and all data processing is performed in main memory. They do provide build-in modules to automatically deal with data bursts, and adaptive resource management across queries. 'Older' data can be pushed to a conventional DBS for longer-term storage.

Data characteristics: DSE are excellent data management tools for real-time analysis of very large numbers of frequently updating streams in applications such as emergency management and manufacturing. Current systems have throughput rates of 1,000,000 updates per second. Real-time data can be co-analyzed with historic data; however, the 'window' of processing is relatively short, for example a window can be few seconds or minutes long, starting from the current time and going back in time for e.g. five minutes. DSE do not support querying any kind of stream intervals involving older data which has expired from the live stream and/or stored in another system.

Systems: TIBCO Streambase [18], Apache/Storm [7], Oracle CQL [20], Microsoft Streaminsight [4, 14], IBM Infosphere [8].

5 Conclusions

Today's geosensor networks produce large numbers of real-time sensor data streams. Sensor data is streamed directly to the cloud or a server for analysis, and users are interested in real-time data analysis. However, such set-ups of continuously streaming sensors bring new data management challenges. In this article, I explored these challenges in more detail, and discussed several options for potential data management support. The first option is centered around Hadoop-based analysis tools, which combine traditional spatial analysis with very

efficient batch job execution using the Map/Reduce paradigm. However, this approach is less useful for real-time analytics. The second choice is traditional spatial and spatio-temporal DBS, which provide declarative data modeling and querying, seamless integration with historic sensor data, and work well for up to 500 updates per second and simple analytical tasks. Thus today, they work well if the data streaming load is capped at about 500 inserts per second. The third option are data stream engines, which have been specifically designed for high-throughput real-time data analysis, and are appropriate candidates for very large numbers of sensor data streams and more complex analytics, providing real-time answers and continuous queries. The need for combing data management support for both high performance batch processing and real-time processing capabilities has led to the investigation of novel hybrid architectures such as the lambda architecture [13]. However, these systems are still under development at this time.

References

[1] J. Abdul, M. B. Potdar, and P. Chauhan. Parallel and Distributed GIS for Processing Geo-data : An Overview. *International Journal of Computer Applications*, 106(16):9–16, 2014.

[2] A. Aji, F. Wang, H. Vo, R. Lee, Q. Liu, X. Zhang, and J. Saltz. Hadoop-GIS: A High Performance Spatial Data Warehousing System over MapReduce. *Proceedings VLDB Endowment*, 6(11):1009, 2013.

[3] I. F. Akyildiza, X. Wang, and W. Wang. Wireless mesh networks: a survey. *Computer Networks*, 47(4):445–487, 2005.

[4] M. Ali, B. Chandramouli, J. Goldstein, and R. Schindlauer. The extensibility framework in Microsoft StreamInsight. In *2011 IEEE 27th International Conference on Data Engineering*, pages 1242–1253. Ieee, Apr. 2011.

[5] AMPLab-UCBerkeley. SPARKR software, 2015.

[6] G. Andrienko, N. Andrienko, P. Jankowski, D. Keim, M. Kraak, a. MacEachren, and S. Wrobel. Geovisual analytics for spatial decision support: Setting the research agenda. *International Journal of Geographical Information Science*, 21(8):839–857, 2007.

[7] Apache. Hadoop, 2015.

[8] A. Biem, E. Bouillet, and H. Feng. IBM InfoSphere Streams for Scalable, Real-Time, Intelligent Transportation Services. In *SIGMOD '10: Proceedings of the 2010 ACM SIGMOD International Conference on Management of Data*, pages 1093–1103, Indianapolis, IN, 2010.

[9] J. Dean and S. Ghemawat. MapReduce : Simplified Data Processing on Large Clusters. In *Symposium on Operating System Design and Implementation (OSDI)*, pages 1–18, San Francisco, CA, 2010. USENIX.

[10] A. Fox, C. Eichelberger, J. Hughes, and S. Lyon. Spatio-temporal indexing in non-relational distributed databases. In *Proceedings - 2013 IEEE International Conference on Big Data*, pages 291–299, 2013.

[11] Github. GIS tools for Hadoop, 2015.

[12] S. Kazemitabar, U. Demiryurek, M. Ali, A. Akdogan, and C. Shahabi. Geospatial stream query processing using Microsoft SQL Server StreamInsight. *Proceedings of the VLDB Endowment*, 3(1-2):1537–1540, 2010.

[13] N. Marz and J. Warren. *Big Data - Principles and best practices of scalable realtime data systems*. Manning Publications, 2015.

[14] J. Miller, M. Raymond, J. Archer, S. Adem, L. Hansel, M. Luti, Y. Zhao, A. Teredesai, and M. Ali. An Extensibility Approach for Spatio-temporal Stream Processing using Microsoft StreamInsight. In *SSTD'11: Proceedings of the 12th international conference on Advances in spatial and temporal databases*, pages 496–501, 2011.

[15] R. N. Murty, G. Mainland, I. Rose, A. R. Chowdhury, A. Gosain, J. Bers, and M. Welsh. CitySense: An Urban-Scale Wireless Sensor Network and Testbed. *2008 IEEE Conference on Technologies for Homeland Security*, pages 583–588, May 2008.

[16] S. Nittel. A survey of geosensor networks: Advances in dynamic environmental monitoring. *Sensors*, 9(7):5664–5678, 2009.

[17] L. Sanchez, J. Galache, V. Gutierrez, J. M. Hernandez, J. Bernat, A. Gluhak, and T. Garcia. SmartSantander: The meeting point between Future Internet research and experimentation and the smart cities. In *Future Network & Mobile Summit (FutureNetw)*, pages 1–8, Warsaw, Poland, 2011.

[18] TIBCO StreamBase. Streambase CEP, 2015.

[19] B. B. Venkatesan. Feasibility study of continuous real-time spatial interpolation of phenomena using built-in functionality of a commercial data stream management system. Technical report, Department of Spatial Information Science, University of Maine, USA, Orono, 2013.

[20] A. Witkowski, S. Bellamkonda, and H. Li. Continuous queries in Oracle. *VLDB '07 Proceedings of the 33rd International Conference on Very Large Data Bases*, pages 1173–1184, 2007.