PD-20 - Real-time GIS Programming and Geocomputation

Streaming data generated continuously from sensor networks, mobile devices, social media platforms and other edge devices have posed significant challenges to existing computing platforms for achieving both high throughput and low latency data processing in addition to scalable computing. This entry introduces a real-time computing and programming platform for time-critical GIS (Geographic Information System) applications. In this platform, advanced streaming data processing software, such as Apache Kafka and Spark Streaming, are integrated to enable data analytics in real-time. This computing platform can also be extended to integrate GeoAI (Geospatial Artificial Intelligence) based machine learning models to leverage both historical and streaming data to achieve real-time prediction and intelligent geospatial analytics. Two real-time geospatial applications in terms of flood simulation and climate data visualization are introduced to demonstrate how real-time programming and computing can help tackle real-world problems with important societal impacts.

Author and Citation Info:

Li, W., Liu, Y., and Wang, S. (2022). Real-time GIS Programming and Geocomputation. *The Geographic Information Science & Technology Body of Knowledge* (1st Quarter 2022 Edition), John P. Wilson (ed.). DOI: 10.22224/gistbok/2022.1.3.

This entry was first published on February 3, 2022. No earlier edition exists.

Acknowledgement

This work is in part supported by National Science Foundation under Grants BCS-1853864, BCS-1455349, GCR-2021147, PLR-2120943, and OIA-2033521 as well as the Laboratory Directed Research and Development Program of Oak Ridge National Laboratory (ORNL), managed by UT-Battelle, LLC, for the US Department of Energy under contract DE-AC05-00OR22725. The US government retains and the publisher, by accepting the article for publication, acknowledges that the US government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this manuscript, or allow others to do so, for US government purposes. DOE will provide public access to these results of federally sponsored research in accordance with the DOE Public Access Plan (http://energy.gov/downloads/doe-public-access-plan).

Topic Description:

1. Introduction

The geospatial world has been increasingly embracing real-time location data. The need for creating smart cities for an improved quality of life has driven the development of IoT (Internet of Things) devices to be deployed in both public infrastructures and smart homes for collecting in real-time information about equipment operations and movement of humans and goods (Li et al. 2020). This way, more informed and timely decisions can be made. For instance, the surveillance cameras on transportation networks could help monitor real-time traffic information for better route planning of commuters. Similarly, devices deployed at airports and other public sites could help improve public safety by using machines to automatically scan and detect abnormal situations from streaming videos in real-time.

Mobile devices, such as mobile phones and GPS-enabled vehicles, have also become a powerful source of real-time geospatial data. In 2020, there are over 5 billion mobile phones being used; 50% of internet traffic comes from mobile devices (Ceci 2021). Each mobile device, as it moves, will generate real-time location and trajectory information that can be used to construct daily activity and mobility patterns of individuals. Using such information, business owners can measure and forecast economic activities through, for instance, an aggregation of customers' visitations to retail stores, the duration of their visits, and the kinds of products that customers are interested in. These datasets could also become important resources for demographic profiling, as patterns of use of mobile phones could be highly correlated with one's socioeconomic status (Blumenstock et al. 2015).

Environmental monitoring and prediction constantly produce real-time geospatial data from sensors and simulations/forecasting. Distributed sensor observation networks, such as the IOOS (Integrated Ocean Observing System), collect data at a very high time frequency. These real-time observational data, depicting the sea surface temperature, wind speed, direction, and water salinity are important inputs for numerical simulation models for weather forecasting and prediction of natural disasters, such as hurricanes (Li et al. 2016). The USGS (US Geological Survey) has also deployed over 850,000 stations which continuously collect surface-water data, including stream levels, stream water, as well as water quality. These are key parameters for predicting flooding events and monitoring water quality for smart water use and treatment, and disaster preparation (Nawyn et al. 2017).

The amount of widely available real-time GIS data and the need for leveraging these data in time-critical applications require new paradigms for real-time geospatial computing and programming. In traditional big data applications, high scalability is often the goal for a high-performance computing (HPC) platform. This means that they need the platform to be able to process large amounts of data that a single computer is not capable of handling due to limitations in memory capacity , hard drives, and/or CPUs (Central Processing Units). In such a scenario, distributed computing is often the solution. Distributed computing relies on distributed computers to tackle a big data problem collectively. Typically, a big data task is partitioned into multiple subtasks, with each computing node responsible for processing a subset of the data. The results from multiple computing nodes will then be collected and aggregated to get the final output. This is also known as a map-reduce mechanism (Li et al. 2014).

Real-time GIS applications, besides high scalability, also require the computing platform to have high throughput and low latency, which is the ability to leverage many computing resources to handle incoming stream throughput and achieve real-time processing of the streaming data. Besides computing platforms, new analytical methods are also desired to extract critical and useful patterns from massive amounts of data. GeoAI or Geospatial Artificial Intelligence, which applies and extends novel machine learning models, such as Convolutional Neural Network (CNN), has become an exciting data-driven paradigm for processing big data and achieving accurate pattern recognition (Li 2020). Its integration with a high throughput computing platform will further empower and improve the real-time capabilities of a GIS system. In addition, visualization is also a critical component in a GIS application to enable visual analytics and to communicate key information to decision makers and the general public. Therefore, enabling real-time visualization of big data through new ways of analytics will further enhance the usability of a real-time GIS application.

The next sections describe fundamentals of real-time GIS and a computing framework for real-time geospatial processing. Emerging and popular tools to support real-time programming and computing are also introduced.

2. Fundamentals in Real-time GIS and Streaming Data Processing

We use the term *geostreaming* to describe data and computing characteristics of real-time geospatial information. Accordingly, a geostream is "a data stream from spatio-temporal data. It is defined by three parts: (1) it is a data stream, (2) it has spatial information, and (3) it has temporal information" (Brandt and Grawunder 2018). In general, a data stream is a continuous ordered data sequence, similar to live audio/video streams, but it often requires more powerful information management and computing paradigms. A data streaming framework assumes that geostream data arrive continuously, are unbounded (endless), and may be out of order. A stream processing algorithm never sees a complete copy of the input stream and can be categorized as an *online algorithm* (Karp 1992). Large and frequent geostreams are often produced and consumed in a distributed computing environment in which data production, management, publishing, receiving, and processing are decoupled as geographically and computationally distributed services.

A data streaming framework is often based on an asynchronous event processing model in which an incoming data stream is discretized as messages, and data producers and consumers interact through a messaging system using message queues (e.g., RabbitMQ) or a publish-subscribe (pubsub) model. Each message has a sequence identifier for ordering. A producer pushes messages to a data stream management system (DSMS). Upon message arrival at the DSMS, an event is created to notify data consumers. A consumer who listens or subscribes to the message via message types or topics then programs an event handler to ingest the message data. Compared to a messaging system in which a message is queued, fetched by a consumer, and removed in a specified order (e.g., first-in-first-out (FIFO) or priority-based), a pubsub model allows a message to be subscribed by multiple consumers and guarantees that every consumer receives a copy of each subscribed messages. Modern data streaming frameworks (e.g., Apache Kafka, Apache Storm, and Apache Flink) often support both message queue-based messaging systems and the pubsub model.

A fundamental question for a data streaming system is how to store and manage unbounded data streams. Data stream management system (DSMS) capabilities can be built into general databases and data warehouses as database extensions (Brandt and Grawunder 2018); DSMSs that are created from scratch are rare. DSMS extensions are broadly available for popular databases, e.g., Oracle Streams, MongoDB Change Streams, and Cassandra Streaming. Database implementation of stream support is centered around the topic of Change Data Capture (CDC), which is a design pattern for how to store, extract, transform, and retrieve incremental data (a.k.a., data change) from incoming data streams. Using timestamp, versioning, or row state are common techniques for CDC. For geostream management, spatial indexing, partitioning, and query algorithms focus on handling the impact of data updates on indexing, access, and query performance and accuracy (Bruns *et al*. 2020). For example, for frequent updates from data streams, updating the index structure, for example *K-d* tree or quadtree, results in a tradeoff between index update timeliness (and thus a performance penalty from frequently computing indexes) and query accuracy (which is affected by stale indexes resulting from less frequent update). CDC for geostreams may use a spatiotemporal data cube to capture and store raw data.

After a partition of geostream data is stored and registered in real-time, it is dispatched for processing by consumers. It is particularly important for a processing function to perform at a speed that matches the flow rate of a continuous data stream because processing logics (i.e., requiring on-the-fly synchronous processing) are often more complex than indexing and storage of stream data (i.e., could be done asynchronously). Parallel processing is thus necessary for geostream data hosted in large databases, data warehouses, and cloud-based data lakes. Scalable data stream processing shares a common pattern with big data analytics models (e.g., *mapreduce*): a processing algorithm takes a subset of input data (data blocks or messages) and computes a partial output. A mapreduce function maps a local function, i.e., *map*, to partitioned data blocks, takes a sequence of key-value pairs as output, and calls *reduce* functions for various filtering, sorting, routing, and aggregation functions, and finally computes a global result or increment.

3. A Computational Framework for Real-time Geospatial Processing

Figure 1 demonstrates a streaming data processing framework that has integrated advanced data analytics capabilities of GeoAI. This framework achieves real-time processing in three phases: sense, reason, and act. "Sense" means data capture and ingestion. A key challenge in handling real-time data from multiple sources, i.e., surveillance cameras, mobile devices, social media platforms, and IoT devices, is a system's ability to ingest data that arrive as streams of events. Addressing this challenge requires a system that possesses high performance, scalability, low latency, and fault-tolerance capabilities. Apache Kafka is a popular choice to build real-time applications because of the strengths mentioned above (Narkhede et al. 2017).

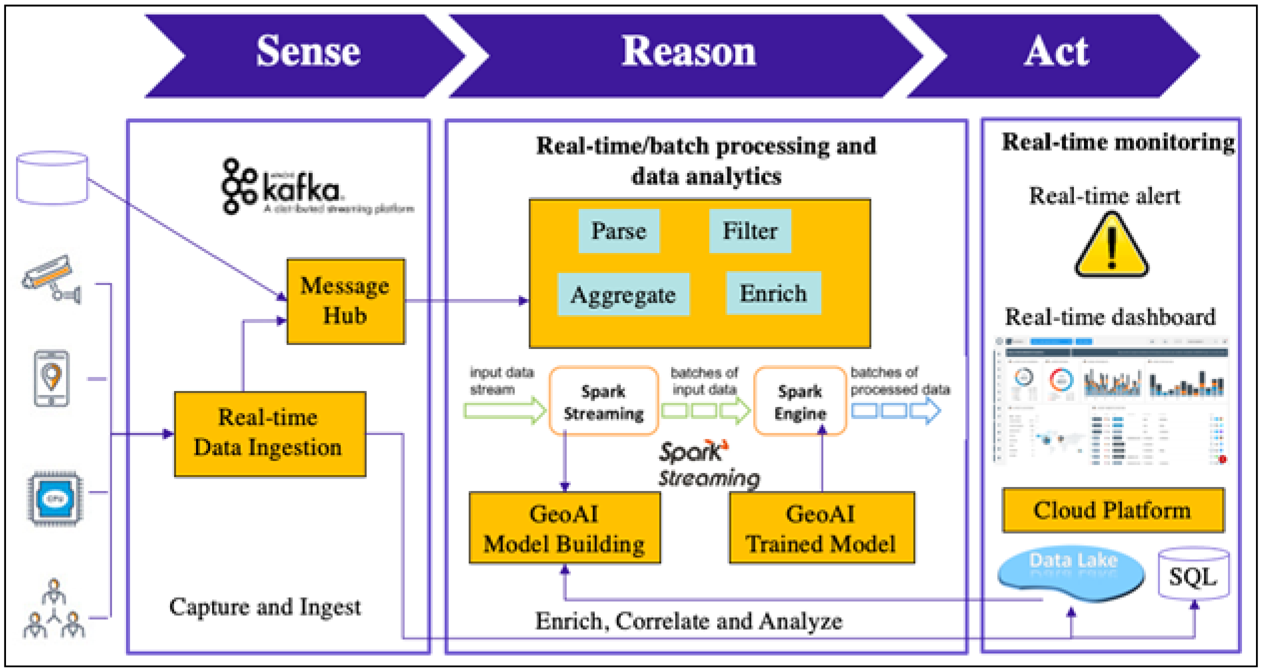

Figure 1. A computational framework for GeoAI empowered streaming data processing (modified from Haddad 2019). Source: authors.

Real-time sensing using Kafka achieves real-time event data ingestion and integration through a distributed computing framework with a highly efficient publish-subscribe messaging mechanism. A Kafka system is composed of servers and clients. A server is a compute node in a cluster responsible for publishing (writing) streaming data into the distributed file system. In Kafka, multiple servers can publish/write the event data in parallel into distributed files or database systems. Its fault-tolerance capability ensures that when a server fails, other servers will pick up the failed server's task and continuously process the data without loss. A client of Kafka works independently from the servers. It subscribes/reads streams of events for further processing. A streaming application can contain multiple Kafka clients to conduct the data read and processing, in parallel. So, the clients themselves can also be distributed and built upon microservices. The communication between a Kafka server and client is through a high-performance TCP (Transmission Control Protocol). The message hub is used for delivering the client services as microservices. Such a service-oriented solution makes Kafka highly flexible, deployable on both cloud and raw distributed systems.

Following the sensing phase, stream data reasoning (Figure 1) often employs real-time data analytics as a stream processing component for improving understanding of the data, through operations such as data parsing, filtering, aggregation, and enrichment. Apache Spark Streaming is a popular tool to process massive amounts of streaming data (Nabi 2016). Spark Streaming extends Apache Spark, a batch data processing framework, and it delivers real-time data processing capability. A Spark stream is created by connecting to a data streaming system, such as Kafka, by subscribing to a topic with a specified stream pulling interval, i.e., the heartbeat interval in Kafka and the batch interval in Spark Streaming. Stream partitions are then organized into Resilient Distributed Datasets (RDDs) in Spark as basic data blocks. These RDDs can be processed using *window* operations, which are designed specifically for stream data processing. For instance, a sliding window, specified by its window length and sliding interval, can be created to prepare traffic time series training data for deep learning-based traffic prediction. Data in a sliding window can be split into the past (samples) and the future (labels). Different sliding intervals would create training data of different temporal resolutions, e.g., timesteps with a minute, hour, or day interval. The processed results can be saved into a more permanent storage, such as a data lake or a relational database, or be pushed to a dashboard for real-time visualization. Various geospatial Spark solutions exist, e.g., GeoMesa for vector processing, GeoTrellis for raster processing, and Apache Sedona, SparkGIS, and LocationSpark for general geospatial indexing, partitioning, storage, and querying. They can be refactored for geostream processing using Spark Streaming. Spatial interdependence is often handled by associating spatial locality in spatial data indexing and partitioning and making it available in the *map* or *reduce* function (Liu et al. 2010).

One exciting facet of the real-time data processing framework is its integration with cutting-edge GeoAI models. New advances in GeoAI, such as the integration of spatial principles (i.e., Tobler's First Law of Geography) with novel deep learning models, have shed new light on spatial data analytics (Li et al. 2021). Because of proven better accuracy in model prediction than traditional analytical models, GeoAI can make the entire data processing and reasoning process more intelligent (Hsu et al. 2021). However, although real-time streaming data provide a rich data source for a GeoAI model to meet its appetite for data, the model training remains a time-consuming process and often cannot meet the real-time requirement. However, using a pre-trained model in a prediction scenario where real-time data are collected can be fast. Real-time prediction, such as vehicle geometry detection by smart cameras at road intersections, only uses a trained GeoAI model, its prediction speed can be magnitudes faster than training the model.

To leverage the strengths of a GeoAI model's high predictability and to achieve real-time performance, a new model training-prediction paradigm can be enabled into the streaming data processing framework (Figure 1). First, the model can be pre-trained and calibrated using historical data stored in the data lake, and streaming data will be used to fine-tune the model in real-time leveraging the high-performance computing resources that Spark offers. Since training is often much slower than the flow rate of geostreams, stream data may first be sampled to select representative data points. Second, the trained model can be integrated into Spark or other parallel computing engines for real-time analytics and prediction. It can also be stored and updated efficiently using edge computing. To further improve the model's prediction speed, pruning techniques can be applied to reduce unnecessary branches without affecting prediction accuracy.

Finally, analytical results become important information to increase situational awareness and for decision makers to reach informed decisions. This is the acting phase in Figure 1. Reasoning results can be visualized in a real-time dashboard to detect anomalies from the streaming data, such as occurrences of disasters or extreme weather, unusual movement of pedestrians, or possible malfunction of a city's smart devices.

Such a real-time geospatial system illustrated in Figure 1 will yield more accurate and rapid prediction and analytics, therefore, supporting timely decisions to improve system efficiency and reduce potential damage/cost (Li et al. 2020). A user-end visualization environment may directly subscribe to geostreams by topics and pull stream data for charting and mapping using JavaScript-based GIS libraries within a web browser. A geostream can also be constantly pushed to a web browser via a WebSocket.

4. Real-time Programming in GIS Applications

In this section, we introduce a general geostreaming service and development tool and two real-world geostreaming applications to illustrate design and programming principles. They each belong to "sense" (Section

4.1), “reason” (Section 4.2) and “act” (Section 4.3) phases of the real-time data processing framework presented in Section 3.

4.1 ArcGIS GeoEvent for Geostreaming

ArcGIS, a leading GIS software program, provides the ArcGIS GeoEvent Server as a solution to real-time event-based geostreaming services. It provides a set of input and output connectors to ingest and publish geostreams into file systems, web, and industry streaming systems, such as Kafka. Real-time analytics on ingested geostreams are enabled by a set of filters and processors in the GeoEvent Server. Filters are used to search and filter spatial features and attributes. Processors are GIS functions that are implemented in mapreduce fashion to produce output event data. A GeoEvent is then published as a stream service and consumed by stream layers using ArcGIS JavaScript libraries. GeoEvent stream services support WebSocket and can be accessed using ArcGIS REST API. Stream clients may consume a stream service using ArcGIS JavaScript, in which the rendering of stream layers may use all data points or only data updates, i.e., track-aware layers.

4.2 Real-time Flood Inundation Mapping and Forecasting

At the national scale, a real-time data analytics approach has been developed to provide flood mapping for emergency management communities (Liu and Sanyal 2020). There are two major data sources: 1) the high-resolution (3m and 1m) digital elevation model (DEM) provided by USGS 3DEP has multiple petabytes and the National Hydrography Dataset (NHDPlus) has 2.7 million river features, covering 5 million kilometers of river reaches; and 2) NOAA’s National Water Model (NWM) is a continuous geostream data source, producing hourly streamflow forecast on the 2.7 million river reaches for the next 18 hours. Figure 2 shows a coherent data-intensive computing framework to handle both big data analytics and large geostreaming using Spark-based microservices. First, spatial indexing and partitioning are built for big raster and vector data management. Rasters are stored as tiles in cloud optimized GeoTIFF (COG) format. Vectors are organized as vector tiles. A raster or vector tile is the basic data block unit. Spark GIS and hydrologic analysis functions are applied in batch computing to produce a hydrologic terrain at the corresponding hydraulic property table. Second, real-time NWM data streams at NOAA are ingested and processed by the Spark Streaming component in this framework to convert streamflow forecast to water depth and flood topology features, such as inundation extent and depth polygons. Finally, these features are published as inundation maps, which are registered as online stream services and can be subscribed in a geovisualization frontend as stream layers using, for example, ArcGIS GeoEvent.

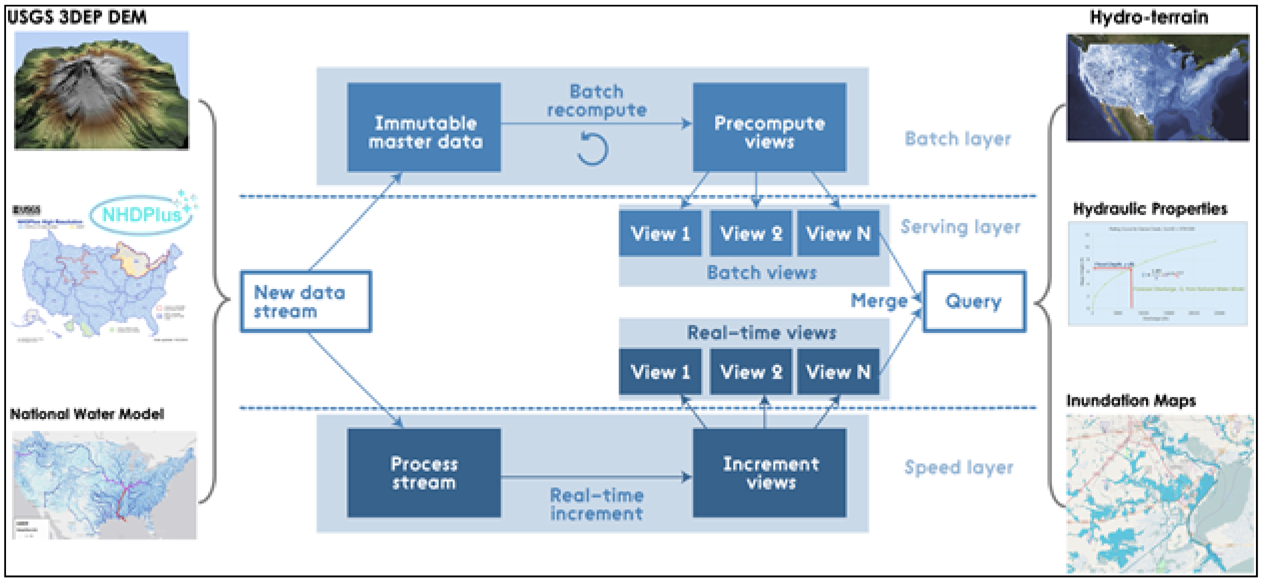

Figure 2. A data-intensive computing framework for high-resolution continental flood inundation mapping and forecasting. The framework (the middle layer) is an abstraction of the lambda microservice architecture. Source: https://dzone.com/articles/lambda-architecture-with-apache-spark.

4.3 Real-time Visualization of Multi-dimensional Climate Data

The world's climate is changing in real-time. Capturing this change through visualization tools will present a unique opportunity for scientists and everyday weather watchers to improve their understanding of climate dynamics and to identify driving factors through the visual aid of some extreme weather events, such as hurricanes or dust storms. Although many scientific visualization tools have been developed, they mostly visualize static data on a layer basis (climate data at a certain pressure level). They lack a mechanism to visualize multi-dimensional, multi-variate, and time-series climate data vividly and efficiently (Li and Wang 2017). To address this challenge, the team at Arizona State University has developed PolarGlobe, a real-time climate data visualization tool that visualizes climate data that are generated from numerical simulation models leveraging advanced computing infrastructure (i.e., cyberinfrastructure; Shao and Li 2019) and high-performance rendering. By developing a parallel visualization algorithm (Wang and Li 2019), PolarGlobe is capable of utilizing client end GPUs (Graphics Processing Units) to render climate variables simultaneously to achieve high rendering speed. Also, through a progressive data transmission strategy, big climate data are encoded into a video stream and transmitted into a visualization client in a very efficient way (Wang et al. 2017). Besides enabling visualization of real-time vector field data (i.e., atmospheric wind) (Figure 3), PolarGlobe also supports the visualization of scalar data in regular and irregular grids.

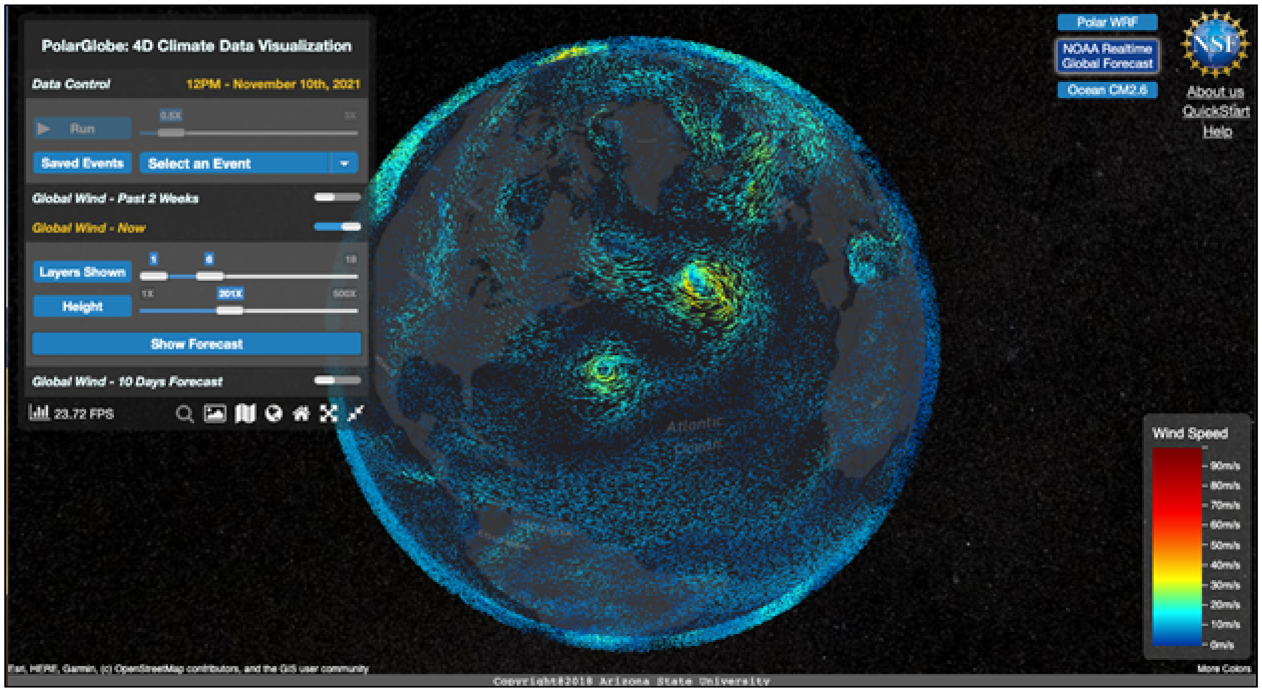
PolarGlobe: 4D Climate Data Visualization
Data Control
12PM - November 10th, 2021
Run
Saved Events
Select an Event
Global Wind - Past 2 Weeks
Global Wind - Now
Layers Shown
Height
Show Forecast
Global Wind - 10 Days Forecast
23.72 FPS
Polar WRF
NOAA Realtime Global Forecast
Ocean CM2.6
NSF
About us
QuickStart
Help
Atlantic Ocean
Wind Speed
90m/s
80m/s
70m/s
60m/s
50m/s
40m/s
30m/s
20m/s
10m/s
0m/s
Esri, HERE, Garmin, (c) OpenStreetMap contributors, and the GIS user community
Copyright©2018 Arizona State University
More Colors

Figure 3. A PolarGlobe visualization interface for atmospheric wind. The real-time data (November 10th, 2021) are from the Global Forecast System (GFS) of NOAA (National Oceanic and Atmospheric Administration). Source: authors.

5. Conclusion

This entry introduces a computing platform that enables streaming data processing and GeoAI enabled real-time data analytics. Such a high-performance computing platform achieves high throughput, low latency, high scalability, and fault tolerance in handling continuous streaming data from diverse sources. In support of time critical applications, several challenges need to be further addressed. First, the data receiver (i.e., Apache Kafka) needs to integrate intelligent mechanisms to filter noisy data on the fly to improve the quality of input data and reliability of analytical results. Second, new techniques are needed to allow GeoAI models to dynamically adjust the weights of historical data and real-time data used in training the model to meet different application needs, i.e., long-term trend analysis vs. near-term, more instant decision making. Third, traditional computing platforms center on distributed computing leveraging multiple CPUs (Central Processing Units). However, for GeoAI models training, GPUs (Graphics Processing Units) remain the more desirable resources to ensure high speed processing. Hence, enabling GPU accelerated computing for GeoAI model training and prediction in a real-time data processing framework would be an important research direction.

References:

Blumenstock, J., Cadamuro, G., & On, R. (2015). Predicting poverty and wealth from mobile phone metadata. *Science*, *350*(6264), 1073-1076.

Brandt, T., & Grawunder, M. (2018). GeoStreams: A survey. *ACM Computing Surveys (CSUR)*, 51(3), 1-37.

Bruns, J., Micklich, F., Kutterer, J., Abecker, A., & Zehnder, P. Spatial Operators for Complex Event Processing. *GI_Forum,* 2020 (2), 107-123.

Ceci, L. (2021). Mobile internet usage worldwide - statistics & facts. https://www.statista.com/topics/779/mobile-internet/#dossierKeyfigures (last access on Nov. 5, 2021)

Haddad, J. (2019). AI Powered Streaming Analytics for Real Time Customer Experience. https://www.slideshare.net/databricks/aipowered-streaming-analytics-for-realtime-customer-experience.

Hsu, C. Y., Li, W., & Wang, S. (2021). Knowledge-Driven GeoAI: Integrating Spatial Knowledge into Multi-Scale Deep Learning for Mars Crater Detection. *Remote Sensing*, *13*(11), 2116.

Li, W. (2020). GeoAI: Where machine learning and big data converge in GIScience. *Journal of Spatial Information Science*, (20), 71-77.

Li, W., Batty, M., & Goodchild, M. F. (2020). Real-time GIS for smart cities. *International Journal of Geographical Information Science*, 34 (2), 311-324.

Li, W., Hsu, C. Y., & Hu, M. (2021). Tobler's First Law in GeoAI: A spatially explicit deep learning model for terrain feature detection under weak supervision. *Annals of the American Association of Geographers*, 1-19.

Li, W., & Wang, S. (2017). PolarGlobe: A web-wide virtual globe system for visualizing multidimensional, time-varying, big climate data. *International Journal of Geographical Information Science*, *31*(8), 1562-1582.

Li, W., Wu, S., Song, M., & Zhou, X. (2016). A scalable cyberinfrastructure solution to support big data management and multivariate visualization of time-series sensor observation data. *Earth Science Informatics*, *9*(4), 449-464.

Li, X., Li, W., Anselin, L., Rey, S., & Koschinsky, J. (2014, November). A MapReduce algorithm to create contiguity weights for spatial analysis of big data. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Analytics for Big Geospatial Data* (pp. 50-53).

Liu, Y. Y., & Sanyal, J. (2020, August). Scalable Data-Intensive Geocomputation: A Design for Real-Time Continental Flood Inundation Mapping. In *Smoky Mountains Computational Sciences and Engineering Conference* (pp. 130-144). Springer, Cham.

Liu, Y., Wu, K., Wang, S., Zhao, Y., & Huang, Q. (2010, November). A MapReduce approach to G i*(d) spatial statistic. In Proceedings of the ACM SIGSPATIAL International Workshop on High Performance and Distributed Geographic Information Systems (pp. 11-18).

Karp, R. M. (1992, August). On-line algorithms versus off-line algorithms: How much is it worth to know the future? In *IFIP congress* (1) (Vol. 12, pp. 416-429).

Nabi, Z. (2016). *Pro Spark Streaming: The Zen of Real-Time Analytics Using Apache Spark*. Apress. 357pp.

Narkhede, N., Shapira, G., & Palino, T. (2017). *Kafka: The definitive guide: Real-time data and stream processing at scale*. O'Reilly Media, Inc..

Nawyn, J. P., Sargent, B. P., Hoopes, B., Augenstein, T., Rowland, K. M., & Barber, N. L. (2017). *User's Manual for the National Water Information System of the US Geological Survey: Aggregate Water-Use Data System, Version 3.2* (No. 2017-1114). US Geological Survey.

Shao, H., & Li, W. (2019). A comprehensive optimization strategy for real-time spatial feature sharing and visual analytics in cyberinfrastructure. *International Journal of Digital Earth*, *12*(3), 250-269.

Wang, S., & Li, W. (2019). Capturing the dance of the earth: PolarGlobe: Real-time scientific visualization of vector field data to support climate science. *Computers, Environment and Urban Systems*, *77*, 101352.

Wang, S., Li, W., & Wang, F. (2017, September). Web-scale multidimensional visualization of big spatial data to support earth sciences—A case study with visualizing climate simulation data. In *Informatics* ,4,(3), 17. Multidisciplinary Digital Publishing Institute.

Learning Objectives:

- Describe characteristics of streaming data and sources of streaming data
- Characterize computing leveraging middleware, such as Apache Spark
- Explain the publish/subscribe messaging mechanism in the server-client based architecture
- Illustrate how Apache Kafka achieves real-time processing of streaming data
- Explain the task for each of the three stages of "sense-reason-act" in a real-time computing paradigm

Instructional Assessment Questions:

1. What are the benefits of and challenges to integrating GeoAI based data analytics into a real-time computing platform?
2. What is the difference between a real-time streaming data processing framework and a general big data processing framework?
3. What time critical GIS applications can benefit from this real-time computing paradigm?
4. How does visual analytics help with spatial decision making?

Additional Resources:

1. Gao, S., Li, L., Li, W., Janowicz, K., & Zhang, Y. (2017). Constructing gazetteers from volunteered big geo-data based on Hadoop. *Computers, Environment and Urban Systems*, *61*, 172-186.
2. Li, W., Song, M., Zhou, B., Cao, K., & Gao, S. (2015). Performance improvement techniques for geospatial web services in a cyberinfrastructure environment–A case study with a disaster management portal. *Computers, Environment and Urban Systems*, *54*, 314-325.
3. Shao, H., Li, W., Kang, W., & Rey, S. J. (2020). When spatial analytics meets cyberinfrastructure: an interoperable and replicable platform for online spatial-statistical-visual analytics. *Journal of Geovisualization and Spatial Analysis*, *4*(2), 1-16.
4. Arundel, S. T., & Li, W. (2021). The Evolution of Geospatial Reasoning, Analytics, and Modeling. The Geographic Information Science & Technology Body of Knowledge (3rd Quarter 2021 Edition), John P.

Wilson (Ed.). DOI: 10.22224/gistbok/2021.3.4.

Related Topics:

- The Evolution of Geospatial Reasoning, Analytics, and Modeling
- Artificial Intelligence Approaches
- Big Data Visualization
- Problems of Large Spatial Databases

Keywords:

- real-time GIS
- Geo AI
- artificial intelligence
- streaming data
- big data
- visualization

ISSN 2577-2848 (online)