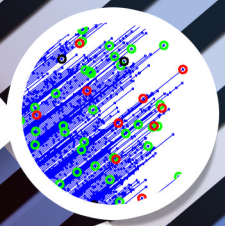

STREAM 2016

Streaming Requirements, Experience, Applications and Middleware Workshop

STREAM2016: Streaming Requirements, Experience, Applications and Middleware Workshop
Workshop Final Report

Office of Advanced Scientific Computing Research, DOE Office of Science
Tysons, Virginia
March 22-23, 2016

This work was supported by the Director, Office of Science, Office and Advanced Scientific Computing Research, of the U.S. Department of Energy. This is LBNL technical report LBNL-1006355.

STREAM2016 is the second of two workshops STREAM: Streaming Requirements, Experience, Applications and Middleware Workshop. This workshop series was made possible by support from the National Science Foundation (Division of Advanced Cyberinfrastructure) and Department of Energy (Advanced Scientific Computing Research) and the Air Force Office of Scientific Research (AFOSR).

Workshop Leads

Geo rey Fox, Indiana Univ.
Shantenu Jha, Rutgers Univ.
Lavanya Ramakrishnan, LBNL

STREAM Steering Committee

Robert Brunner, UIUC
Dennis Gannon, Indiana Univ.
Maryam Rahnemoonfar, Texas A& M
George Djorgovski, Caltech
Shrideep Pallickara, Colorado State Univ.
Madhav Marathe, Virginia Tech

STREAM2016 Organizers

Amy Apon, Clemson Univ.
Scott Klasky, ORNL
Shrideep Pallickara, Colorado State Univ.
Katherine Riley, ANL
Carlos, Varela Rensselaer Polytechnic Inst.
Dennis Gannon, Indiana Univ.
Kerstin Kleese van Dam, BNL
Manish Parashar, Rutgers Univ.
Craig Tull, LBNL
Matthew Wolf, Georgia Tech

Workshop Participants

Gagan Agrawal, Ohio State Univ.
Frederica Darema, AFOSR
Chen Ding, Univ. of Rochester
Salman Habib, Argonne National Lab
Marty Humphrey, Univ. of Virginia
Daniel Katz, National Science Foundation
Shweta Khare, Vanderbilt Univ.
Andre Luckow, Clemson Univ.
Andre Martin, Dresden Univ. of Tech.
Klaus Mueller, Stony Brook Univ.
Stacy Patterson, Rensselaer Polytechnic Inst.
Brian Quiter, LBNL
Rahul Rao, U.S. Airforce AFMC, AFRL
Martin, Swany Indiana Univ.
Nathan Tallent, PNNL
Venkatram Vishwanath ANL
Dean Williams, LLNL
Shinjae Yoo, BNL
Roger Barga, Amazon
Jack Dennis, MIT
John Harney, ORNL
Graham Heyes, Je erson Lab
Dimitrios Katramatos, BNL
Raj Kettimuthu, ANL
Eugene Kirpichov, Google
Madhav Marathe, Virgina Tech
Nina Mishra, Amazon
Srinivasan Parthasarathy, Ohio State Univ.
Je Porter, LBNL
Karthik Ramasamy, Twitter
Mohsen Amini Salehi, Univ. of Louisiana
Alexander Szalay, Johns Hopkins Univ.
Vakho Tsulaia, LBNL
Patrick Widener, SNL
Kesheng (John) Wu, LBNL
Dantong Yu, BNL

Contents

Executive Summary

The Department of Energy (DOE) Office of Science facilities including accelerators, light sources, neutron sources and environmental sensors are producing large volumes of streaming data. The streaming data needs to be analyzed in reactive real-time, or processed in near real-time to enable next- generation scientific discoveries. There has also been an explosion of new research and technologies for stream analytics from the academic and private sectors to address the growing data volumes from social media and other web applications. However, there has been no e ort in either documenting the critical research opportunities or building a community that can create and foster productive collaborations across scientific disciplines, government agencies and industry. To address these shortcomings, a two-part workshop series, STREAM: Streaming Requirements, Experience, Applications and Middleware Workshop (STREAM2015 and STREAM2016), was conducted to bring the community together as well as to identify gaps and future e orts needed across various funding agencies.

This report describes the discussions, outcomes, and conclusions from STREAM2016: Streaming Requirements, Experience, Applications and Middleware Workshop – the second workshop in the STREAM series, held on March 22-23, 2016 in Tysons, VA. STREAM2016 focused on DOE applications, computational and experimental facilities, as well as software systems. The role of streaming and steering as a critical aspect of the linkage between experimental and computing facilities was pervasive throughout the workshop. Given the overlap in interests and challenges faced by industry, the workshop had significant presence from several major companies in this area.

The workshop identified the importance of supporting streaming to meet the requirements of DOE science applications. The rate and volume of data that is being produced indicates an urgent need to systematically approach the gaps and challenges in managing and processing streaming data. Streaming needs are expected to a ect the entire scientific software ecosystem. The community needs to research and develop tools and technologies, including those informed by current commercial systems, that will both work across the scientific application landscape and go beyond the ad hoc solutions that are available today.

This report discusses four research directions identified by STREAM2016 as important for current and future application requirements. The R&D areas include: (i) Algorithms, (ii) Programming Models, Languages and Runtime Systems, (iii) Human-in-the-loop and Steering in Scientific Workflows, and (iv) Facilities. Additionally, the workshop identified a need for community development activities including development of benchmarks, evaluation of existing industry solutions, education and training, and conferences/workshops to explore streaming challenges. The identified research directions provide an important opportunity for building competitive research and development programs around streaming data. The findings and recommendations of this report are consistent with the vision outlined in the NRC Frontiers of Data and National Strategic Computing Initiative (NCSI) [1, 2].

Findings & Recommendations

Findings

1. **Many DOE facilities and projects have critical streaming and steering needs.** The workshop identified many streaming and steering needs from the DOE Office of Science facilities and projects represented at the workshop. Many scientific experiments involve multiple instruments, HPC simulations or data analysis steps, large volumes of data and distributed collaborators. Thus, workflows that involve streaming and steering of large data movement using heterogeneous distributed set of resources and research specialists need to be supported.
2. **The requirements of DOE and other applications need to be more systematically studied to enable generalized solutions in the future.** Currently there is no agreed upon classification of requirements and solutions. There is a need to study the requirements of streaming applications and compare current domain-specific, open-source and industry software environments and identify gaps.
3. **The advances in sensing technology and computing power that are available today require innovations in streaming algorithms, applied mathematics, and statistics.** Existing work in algorithms, mathematics and statistics for streaming systems is not yet mature, with several areas where improvements are needed and with many open challenges. There has been initial research in online algorithms whose complexity is linear in number of data points, but these algorithms need to be refined and turned into usable libraries.
4. **Addressing streaming challenges will require investigation into existing and new programming models.** The programming and runtime requirements of streaming applications are di erent from the traditional MPI-based applications that have dominated scientific computing. As such, there is a gap in capabilities of existing programming models. There is a need for a systematic evaluation of existing approaches and research in new programming models.
5. **Managing the end-to-end workflow is critical to ensure innovations from streaming data.** The complexity of the end-to-end orchestration of real-time streaming data and processing imposes performance constraints and new functional requirements. There is a critical need to support real-time steering and human-in-the-loop activities for next- generation streaming workflows.
6. **Meeting the needs of the streaming and steering applications will require development of capabilities and support from DOE high performance computing and networking facilities.** Historically, DOE HPC facilities have focused on batch queue jobs. Also, the network is a critical part of the streaming and steering dataflows, but is not seamlessly integrated into user workflows. There is a need for better technical support and policies that facilitate streaming data and steering jobs with batch capabilities.
7. **There is a need for community e orts and infrastructure to develop and support capabilities for streaming.** It was recognized at the workshop that organized community activities for streaming are

limited. There is a need for community-building activities that include engagement across scientific domains, academia, and industry.

Recommendations

1. **Develop new algorithms to support data streaming and steering needs.**
   (a) There is a need for research into online algorithms, which use ideas such as sampling and sketches, to support streaming and reduce their time complexity. Metrics to assess the applicability and quality of these algorithms are needed.
   (b) Machine learning algorithms including deep learning, graph analytics, and dimension reduction, need to be extended to a broader range of DOE applications. Their value and implementation needs to be evaluated for functionality and performance.
   (c) There is a need for scalable parallel versions of existing streaming algorithms and frameworks that can scale current algorithms (e.g., machine and deep learning) to large data sizes.
2. **Evaluate existing and research new streaming programming models that address hardware and application challenges.**
   (a) There is a need to define and compare application requirements and existing programming models with respect to their functionality, performance, and usability for streaming applications.
   (b) There is a need to understand the relation between application workflows, streaming programming models, and execution models, and the consequences for performance and programmability.
   (c) There is a need to explore the impact of emerging technologies such as new memory and processor technologies, and new storage ideas such as object stores.
3. **Enable support for end-to-end streaming dataflows while enabling human interactivity and steering.**
   (a) Engage with DOE facilities and projects using human computer interaction (HCI) methodologies to build a deeper understanding of the usage models and gaps in the end-to-end workflow characteristics. There is a need to develop classifications of usage models that can feed into benchmarks and mini- applications focused on streaming and steering needs of the applications.
   (b) Develop capabilities and QoS metrics to evaluate the end-to-end workflows in data streaming to address the usability, robustness, and performance needs of streaming applications.
   (c) Develop advanced methodologies to allow for composition and management of stream processing pipelines.
4. **Adapt HPC computational and networking facilities' services to address the needs of streaming and steering.**
   (a) Investigate and develop scheduling policies and algorithms that enable stream processing concurrently with batch jobs and/or the co-scheduling of distributed resources. Similarly policies and algorithms to support interactive stream processing are needed.
   (b) Explore existing and emerging data streaming technologies (e.g., Spark, Heron) in HPC facilities and develop solutions that enable them to be integrated with existing software stacks and infrastructure.

(c) Expand on the capabilities of the ScienceDMZ architecture to enable analytics and processing on the wire for data streams.

5. **Develop and sustain a software ecosystem that supports the needs of the Office of Science applications.**

   (a) Develop benchmarks and mini-applications that can be used to evaluate existing solutions.

   (b) Identify community best practices and drive generalization across science needs by developing common libraries.

   (c) Develop techniques and methodologies that enable streaming solutions to be integrated into existing domain-specific software stacks and ensure long-term software sustainability.

   (d) Bridge the research versus product tension, and allow for robust prototypes to be integrated into existing application/domain-specific software stacks.

6. **Develop and build a community around streaming and steering within DOE and across federal agencies.**

   (a) Create a community across federal funding agencies to develop, discuss, and manage R&D associated with streaming and steering.

   (b) Facilitate demonstration projects that integrate experimental-observational facilities with HPC systems for production streaming pipelines so as to stress test computational facilities and existing software systems.

   (c) Develop support for streaming applications including cases with a single or multiple data sources and a range of event sizes.

1

Introduction

The national, energy, and economic interests of the United States are increasingly dependent on the ability to analyze the data from experiments, observations and simulations of complex phenomena from DOE Office of Science facilities. In the last few years, the volume and complexity of data being produced has been growing exponentially and there is an increasing need for real-time processing of the data as it is being produced. Thus, there is a growing interest and urgent need to address streaming data and real-time steering and control from on-line instruments, large-scale simulations, and distributed sensors.

The goal of the STREAM workshop series was to identify the applications, state of the art, gaps and challenges that need to be addressed by future R&D e orts. STREAM2015, held in Indianapolis, was the first of two workshops and had a focus on NSF applications and infrastructure [1]. The goal of the second workshop, STREAM2016, held in Tysons, VA, was to understand needs related to streaming, and related steering and control in DOE science applications.

STREAM2016 workshop participants discussed the terminology to understand the scope of streaming and steering and expanded on the definitions from STREAM2015. We present the working definitions of Stream and Steering used at the workshop.

Stream. A stream was defined to be an unbounded sequence of events that needs reactive real-time or near real-time processing and can possibly have multiple consumers and producers. Successive events may, or may not be correlated and each event may optionally include a time-stamp. Challenges arise as streams do not support random access, i.e., arbitrary forward or backward access to data, but need to be processed as it arrives, thus cannot benefit from global meta-data awareness.

Steering. Steering is defined as the ability to dynamically control the progression of a computational process to enable decision support. Steering which is inevitably real-time, might include changing the progress of simulations, or realigning experimental sensors or instruments, or control of autonomous vehicles. Streaming and steering often occur together, though that is not always necessary as in the case of two coupled codes (e.g., two solvers). Steering is often also tightly driven by the human-in-the-loop. It was also noted, that the time scale of steering could impact the steering process utilized. For example, steering on the microsecond time scale may need fully automated methods, while steering on the 10s to 100s of seconds time scale could have a human-in- the-loop.

This report summarizes the findings from the second workshop, STREAM2016, for Advanced Scientific Computing Research (ASCR) and the research community. STREAM2016 received 27 white papers. Given the importance of building a community and understanding the broad range of viewpoints, we elected to invite all submitted white papers. The workshop had 49 attendees and consisted of invited talks, presentations of the white papers, discussions after each session, and two breakout sessions. The presentation and breakout sessions were moderated by community members. Reflecting the broad scope, the submitted

white papers covered a mixture of applications, technologies, and state of practice in the science communities, academia, and industry.

STREAM2015 Summary. We summarize the STREAM2015 findings to set the context for the rest of this report. STREAM2015 had 43 attendees, 17 workshop white papers (from call for participation) and 29 presentations (28 with slides; 23 with videos). The workshop covered the field broadly and there was a consensus that it usefully brought together an interesting set of participants and there was enthusiasm for continuing such activities. The workshop covered technology, applications and, education and included industry participation from Amazon, Google, and Microsoft (technologies) and Johnson Controls (Industrial Internet of Things IIoT). The application discussion given in Sections 2 and 3.1 was initiated.

Current technology solutions were reviewed with a plethora of local point solutions but few end-to-end general streaming infrastructures were available in the non-commercial space. There were a range of different open-sourced big data systems such as Apache Spark [2, 3], Flink [4], Storm [5], Heron [6], Samza [7], and the proprietary Kinesis [8] and MillWheel [9]. However, the majority of these solutions targeted needs that are di erent from those of DOE applications. Issues identified included the need to optimize current XSEDE and DOE infrastructure for streaming data and the need to address several issues of importance in distributed computing, such as performance, fault-tolerance, and dynamic resource management. The interface of HPC and streaming was extensively discussed. Novel algorithm issues including online and sampling methods and the reduction of $O(N^2)$ algorithms to $O(NlogN)$ were highlighted. The importance of benchmarks, application collections and streaming software system and algorithm libraries were stressed, as was the need for infrastructure that is optimized for streaming applications.

The rest of this report is organized as follow. We summarize the use cases in Section 2 and describe the state of the art in Section 3. We detail the R&D challenges identified at the workshop in Section 4, the community activities in Section 5, and summarize the workshop in Section 6.

2

Application Use Cases

In this section, we capture numerous application use cases from various DOE scientific domains that were represented at the workshop. There are additional DOE applications that have streaming needs. Along with the main DOE applications, the workshop talks also covered sensor-controlled nanotube growth [P16-16, WP16-16], control systems for aircrafts [P16-22], video streaming [P16-2, PWP16-2], urban population data, and stock market data. Designing and building generally applicable systems, will require us to understand the needs, gaps, and challenges of applications across a broad range of applications inside and outside DOE.

2.1 High Energy Physics Experiments

High Energy Physics (HEP) and particle physics experiments, such as ATLAS, CMS, Daya Bay, or LZ Dark Matter Experiment, routinely run in a steady-state, data-taking mode over many years, collecting very large volumes of data [P16-23]. This data consists of a continuous or punctuated stream of discrete events associated with elementary particle interactions and/or triggered detector data readouts which can be treated as an atomic data unit. The data streams are often handled and processed in near real-time for the purpose of validating detector health and data integrity. Data streams are also saved to files and curated for subsequent offline calibration and processing to extract the maximum scientific yield from the experiment.

The ATLAS LHC experiment has 3000 scientists and 1200 students from 38 countries [10]. It gathers 1 PB/sec of data filtered to 1-2GB/sec recorded. The experimental computing environment is instantiated on 140 heterogeneous worldwide resources enabled by excellent networking. ATLAS data processing is moving from classic file-based grids to an Event Service (ES) [11], which is operational today. The infrastructure supports quasi-continuous event streaming through worker nodes and exploiting opportunistic resources and minimizing local storage demands. ATLAS uses the PanDA Distributed Workload Manager [12] to manage processing on HPC systems. It features whole-node scheduling, remote I/O, and use of object stores. The need for real-time processing requires that the infrastructure handles real-time, time-sensitive stream processing.

Belle II at KEK in Japan gathers 25 PB/year of raw data [13]. Belle II has hierarchical scheduling to mitigate contention and reduce power consumption. This uses a sophisticated analytical model to predict execution time fed by a provenance engine to gather performance metrics. Today, data transfer is optimized by prefetching. Future needs for Belle II indicate a need for stream processing.

In general, HEP experiments typically have higher throughput requirements for processing their data streams than other experiments such as those at Light Sources. HEP experiments' data streams over months

and years constitute a single dataset which is regularly revisited for higher-statistic and better- calibrated analyses and, consequently, high-quality scientific results. In this sense, the HEP data "stream" is replayed for offline analysis — but in a much shorter amount of time. For example, the entirety of four years of Daya Bay [14] data is re-streamed through the data management, processing, and analysis within a short one-month reprocessing campaign. Typical practice is to use the same dataflow and workflow machinery to process the offline data stream as is used for online, real-time streaming processing.

2.2 Light Sources

Light Sources are facilities with scores of di erent beamline end-stations (in the case of synchrotrons) that serve thousands of researchers each year from many science domains (chemistry, biology, material science, archaeology). These end-stations provide the researchers a wide variety of capabilities and X-ray energies suited to answering many di erent scientific questions of interest to the researchers. Hence, the facility generates many independent streams of data, each with it's own characteristics and scientific use. Each end-station has a sequence of end-users and experiments typically consisting of a number of related samples being studied as an ensemble. The atomic data unit for a Light Source end-station is typically a single sample. In-situ experiments then aggregate those atomic data units into a data time-sequence.

X-ray Light Source data, once taken by a particular experimental team, are treated as discrete datasets and analyzed independently without coordination with analysis of other datasets or experimenters. However, an emerging concept of common data repositories has been developing at some Light Source facilities. In this concept, datasets are not aggregated for analysis, but metadata searches and comparisons may yield important correlations between experiments conducted by separate scientists.

Stream processing is a goal of many Light Source software systems such as SPOT Suite at the Advanced Light Source [15, 16]. Automated data management, processing, and curation are becoming standard demands of Light Source end-users. An X-ray Light Source data stream with slow streaming feedback may invalidate an entire set of experiments, sending a researcher home with no analyzable data. This leads to greater emphasis on near real-time, streaming analysis with low latency for Light Sources.

2.3 Ameriflux Network Management Project

The AmeriFlux network is a community of sites with sensors and scientists measuring ecosystem carbon, water, and energy fluxes across the Americas. The datasets provide critical linkage between organisms, ecosystems, and process- scale studies of climate-relevant artifacts such as landscapes, regions, and continents. In turn, these data products can be incorporated into biogeochemical and climate models. When viewed as a whole, the network observations enable multi-resolution study of trace gas (CO_2, water vapor) fluxes across a broad spectrum of times (hours, days, seasons, years, and decades) and space. AmeriFlux observations have been instrumental in defining the relationships between environmental drivers and responses of whole ecosystems, which can be refined using machine learning methods like neural networks or genetic algorithms informed by remote sensing products.

AmeriFlux is an example of an Internet of Things scenario, where there are many sensors out in the field that create atmospheric measurement records of varying quality and heterogeneous formats. The goal is to move from sporadic manual data submissions to an automated and streaming data collection, cleaning and analysis environment. This will shorten the time from data collection to data product delivery for the scientific researcher. Recently, the Ameriflux team has been experimenting with using public cloud services including Amazon Kinesis and Lambda for collecting and processing environmental data streams [P16-6].

2.4 Radiological Search

Sensors deployed on aircrafts (ARES) and trucks (RadMAP) collect ambient gamma-rays and neutron measurements to detect, localize, and identify possible threats against a background of benign radioactivity. ARES streaming algorithms have been developed to highlight spectral changes or matches to threat signatures [17]. The greatest challenges in developing these algorithms is the low statistics available and the naturally changing radiological environment. Current radiation sensors produce data at 500GB/hr after online processing and the video cameras produce data at 2TB/hr. RadMAP data is currently only processed after collection [18].

Investigations are now underway on an improved streaming analysis of this data with information from other contextual sensors, including high-definition video, Lidar, and Hyperspectral imagery being integrated in real time, leading to a streaming multi-source, heterogeneous data type scenario [P16-20]. Furthermore, there is a desire to test if computationally steered measurements based on the streaming analysis would lead to more efficient data gathering and more e ective detection.

2.5 Combustion Science

The goal of combustion science experiments is to understand the dynamics of fuel mixes, speeds and acoustic interactions. Particles are injected into the combustion and sets of cameras capture images at time intervals, allowing velocity fields to be calculated [P16-27,WP16-23]. The streaming data are the images which are processed first to determine if the data is valid, failing which, the experiment must be rerun. If the data is determined to be valid, the next step is usually to identify features in physical parameter space that are of interest. If nothing of interest is found, then the experiment must be refined and the process is repeated. In addition to experimental streaming data, cross-stream computations that relate the experiment to previous runs and simulations are performed. Additionally, collaboration between multiple sites with relevant data must be included in the analysis. The current methods are traditionally based on ad hoc software that was designed to fit the very specific needs of the experiments. There is a growing need for generalized methods to process the stream data.

2.6 High-End Electron Microscopy Experiments

This involves the steering of high-end electron microscopy experiments where a beam of electrons is transmitted through an ultra-thin specimen. These experiments can generate atomic resolution di raction patterns, images, and spectra under wide ranging environmental conditions. These experiments generate from 10GB to tens of TB (e.g., at Brookhaven National Laboratory) of data at rates ranging from 100 images/sec for basic instruments to 1600 images/sec for state-of-the-art systems [P16-13,WP16-13].

The current workflow management systems in use include the Analysis in Motion framework [19] developed by Pacific Northwest National Laboratory. There are open challenges in reliably executing the workflow, that consist of composite applications built from loosely coupled parts, running on a loosely connected set of distributed and heterogeneous computational resources. Each workflow task may be designed for a di erent programming model and implemented in a di erent language, and most communicate via files sent over general purpose networks.

To optimize the scientific outcome of the microscopy experiments, it is required to analyze and interpret the results as they emerge. It is essential that the workflow management system reliably deliver optimal performance, especially in situations where time-critical decisions must be made or computing resources are limited.

2.7 International Fusion Energy Projects

International fusion energy projects such as the International Thermonuclear Experimental Reactor (ITER), Korea Superconducting Tokamak Advanced Research (KSTAR), and National Spherical Torus Experiment (NSTX) have streaming needs [P16-12,WP16-12]. In these projects, teams of scientists are present at the facilities to monitor the progress of the on-going data collection, adjust the control settings, and prevent catastrophic events, while most others access the data remotely. These projects thus represent collaborative data analysis challenges. A specific important challenge is handling the growing data sizes over the network.

There is a need for a software system that allows scientists to quickly and conveniently compose complete analysis tasks, manage the necessary data movement, execute the specified tasks, and provide timely feedback to the users.

2.8 Astronomy

The Sloan Digital Sky Survey (SDSS) produced 100TB of processed data from 1992 to 2008. Such large data sets are just the hint of the future with more data coming from both fewer telescopes and large simulations presenting major challenges. It is important to realize that only O($NLogN$) algorithms or better, are realistic to process current and future data sizes. Systematic errors are more important than statistical errors. SDSS uses approaches based on streaming and sampling aimed at robust techniques for dimensional reduction with a streaming PCA (Principal Component Analysis). Time domain data is of growing importance and requires fast triggers. Streaming algorithms have already been shown to be beneficial in this community, e.g., halo-finding (identification of a gravitationally bound objects) has shown huge memory gains from the use of streaming algorithms. STREAM2015 covered the Square Kilometre Array (SKA) project [P15-6] with its dramatically large data volumes.

2.9 Earth System Grid Federation

The Earth System Grid Federation (ESGF) is a coordinated multiagency, international collaboration of institutions that continually develop, deploy, and maintain software needed to facilitate and empower the study of climate [20]. ESGF relies on real-time data movement across the sites. The immediate goal of the collaboration is 4 Gbps (1 PB/month) of sustained disk-to-disk data transfer between ESGF primary data centers. The stretch goal is to be able to sustain 16 Gbps (1 PB/week) of sustained disk-to-disk data transfer between ESGF primary data centers. Stream processing in the ESGF and related collaborations focus on visualization and analyses where data is remapped to a common domain and in some cases large domains might be down sampled. The community uses an interactive, view-dependent data loading method where fast coarse resolution is used for quick results and user-directed high resolution visualization is produced.

2.10 Cosmology

Streaming dataflows and associated analytics play an essential role in cosmology. Streaming data in cosmology come from various sources including CMB (Cosmic Microwave Background) experiments, optical transients, radio surveys. Stream processing in cosmology include in situ and co-scheduled data transformation in simulation pipelines and analytics that include transient classification and imaging pipelines. Optical

searches for transients, e.g., Dark Energy Survey (DES), Large Synoptic Survey Telescope (LSST) and Palomar Transit Facility (PTF) can have cadences in the range of fractions of minutes to minutes. Current data rates are typically about 500 GB/ night. LSST can go up to 20TB/night, about 10K alerts/night.

There are major opportunities for online machine learning for filtering and classification of transient sources. A major challenges for machine learning approaches in Cosmology are high levels of throughput and lack of training datasets. Additionally, there is often need for multiple access through streaming data, need for management of the complex software pipelines, and on-demand allocation of resources.

2.11 Power Distribution Grid Monitoring

The power distribution grid monitoring project uses high-resolution, micro-phasor measurement units (uPMUs) to design and implement a measurement network, which can detect and report the resultant impact of cyber security attacks on the distribution system network. The project is developing a framework to directly measure, in real time, the physical state of the distribution network at many points and then compare the actual measured data from the distribution grid to the system's state as reported by cyber monitoring. The result will be a system that provides an independent, integrated picture of the distribution grids' physical state, which will be difficult for a cyber-attacker to subvert using data-spoofing techniques.

Regardless of detection of suspicious events or not, devices send the resulting data to a central location for further analysis, either directly or via a tree structure that could also perform additional analytics at each layer along the way depending on whether doing so would improve or hinder detection accuracy and latency in the desired way. Currently, the project is leveraging combinations of RabbitMQ and the Broker client communication framework that is part of the Bro Network Security Monitor for messaging, as well as the Cassandra database to store the time-series data. Thus, the power grid monitoring analysis infrastructure requires a stream processing and storage framework that can scale and process thousands of events.

This project is illustrative of other important electric grid activities and more generally the IoT (Internet of Things) area.

2.12 Monitoring data at ASCR Facilities

Another important use case comes from monitoring data that needs to be processed in streaming mode at ASCR Facilities [WP16-10,WP16-24] ASCR facilities including HPC centers and ESnet, continuously monitor metrics on thousands of devices. This data is processed as streaming time-series using the current state-of-the-art technologies, and analyzed for both usage and utilization statistics as well as anomaly detection, debugging, and performance tuning.

For example, NERSC has recently built up a cluster of over a dozen nodes to monitor many aspects of the Edison and new Cori supercomputers. The infrastructure uses RabbitMQ for routing messages coming from multiple locations in the supercomputing center, and ElasticSearch for real-time indexing, reporting, and data storage. Analysis functions such as streaming alerts can be built either on ElasticSearch or directly on RabbitMQ queues. The system handles millions of individual data points per hour and is built to move data from fast short-term storage, through spinning disk, to long-term archival. Also, ESnet has several network monitoring systems currently collecting and handling a large volume of monitoring data at all levels of the networking stack. For example, ESnet runs the esmond software [21], a system for collecting, storing, visualizing and analyzing large sets of time-series data. Esmond stores raw data to Cassandra while performing on-the-fly summarization, and puts descriptive metadata in SQL for easier ad hoc querying.

The ESxSNMP system [22] performs similar functions for ESnet Simple Network Management Protocol (SNMP) counters from across the network.

HPC Centers have built up experience collecting, analyzing, and storing large quantities of sometimes heterogeneous streaming data. HPC centers have built up expertise in managing large and scalable deployments of mainstream commercial streaming data technologies such as RabbitMQ, ElasticSearch, and Cassandra, that will be critical as we build a streaming software ecosystem.

3

State of the Art

In Section 2, we summarized key use cases and the importance of streaming for DOE facilities and scientific applications. We also highlighted future needs, gaps and challenges that streaming presents for these use cases. In this section, we provide a summary of the state of the art of the field across academia and industry.

An analysis of the applications in Section 2 yields several observations and areas of further examination.

- There is a range of experimental facilities and observational systems with a variety of streaming data and computational characteristics, such as National Synchrotron Light Source (NSLS-II), APS and ALS to name just a few.
- There is a wide range of streaming data types: from a variety of experimental data in di erent formats [P16-24], images [P16-27], and non-real time collision events [P16-25].
- Requirements of the temporal response of the end-point processing vary significantly:
  - O(10^5) seconds [Overnight]: Plan campaign for next shift/day (e.g., telescopes, day shift experiments).
  - O(10^4) seconds [Hourly]: Detect problems; Maintain steady-state data taking (e.g., stable, long-term HEP experiments).
  - O(10^2) seconds [Minutes]: Follow experiment evolution and verify data quality (e.g., time-resolved, in situ experiments).
  - O(1) [Instantaneous]: Systems that require real, or near-real time processing (e.g., cloud-linked robots and autonomous vehicles).
- Often there is a reduction in the volume (rate) of data streamed. For example, in HEP experiments the volume (rate) of data that is generated is not always the same volume (rate) as the data that is streamed.
- Distributed resource management techniques are currently primitive, and are mostly carry-overs of HPC static resource management techniques [WP16-14]
- There is a need for unified model for batch and stream data processing. There are several ongoing e orts, e.g., Apache Beam [P11-16], Flink, Spark, Pilot-Streaming [P16-14].
- Current software solutions are mostly based on ad hoc and bespoke software. Unification across semantic, syntactic and representational di erences between data streams is needed, as well as across

Application Class	Application Features	Details and Example	Talks and White Papers
Data Assimilation	One-way integration of typically distributed data, into real-time computation process.	Defined source and sink endpoints.	WP 16-6
Computational Steering	Integrate distributed data with computation; determine next-steps of simulations based upon results. Typically a single control loop.	Computational steering also include in-* analysis (e.g., in situ and in-transit). Experiments can steer computation, and vice versa.	WP 16-20, WP 16-16
Distributed and Integrated Computations and Experiments (DICE)	High-performance and distributed workflows comprised of many concurrent execution paths. Data is typically derived from a single source even if there are multiple points of processing.	Currently solutions typically integrate batch-stream and processing. Large experiments (e.g. Light and neutron sources, etc) and observational systems (e.g, Astronomy).	P 16-25, P 16-5, P 16-24, P 16-29 WP 16-9, WP 16-12, WP 16-13, WP16-23
Concurrent Multi-stream Applications	Many streaming events, often with real-time response needed. Typically, many independent data sources. Also, small independent events and distributed analysis.	Internet of things (IOT), Cyberphysical, DDDAS, and Social Media, Satellite and airborne monitors requiring image analysis.	P 16-11 WP 16-12

Table 3.1: A high-level classification of the types of streaming applications.

scales. There is a case to be made for the unification of HPC and commercial software stack [P16-1].

- There is a need for streaming versions of existing and new algorithms. These range from data compression, sampling, and analysis on compressed data streams, to machine learning for anomaly detection and other problems [P16-22].

3.1 Application Classes

We observe that although streaming applications di er widely in the type of science they support (simulations versus experiments), a large fraction of streaming applications can be understood using just a few common characteristics, such as, whether they are distributed or not, the size and type of streamed data unit, whether they have real-time constraints and their end-point resource requirements. Using these common and important characteristics, we propose four high-level application classes.

This is illustrated in Table 3.1, which provides a coarse-grained classification[1] of streaming applications and their primary features. These classes are not mutually exclusive: an application might have features that are drawn from multiple classes.

Irrespective, it is critical to understand the characteristics of the data streaming aspects of the workflow: (i) What is the role or need for streaming?; (ii) What type of data and information is streamed?; (iii) How do current applications support streaming?; and (iv) Gaps, requirements, and state of streaming.

The lack of clarity and certainty about the number and types of classes of streaming applications and scenarios, points to the need for mini-apps (often also called skeletons) and benchmarks for streaming and steering. The mini-apps will help to characterize some areas and facilitate a discussion on why things are di erent.

[1]The classification is not ”rigid” or ”exclusive” but mostly illustrative of primary properties.

Several features could be used to develop a more detailed classification, or to derive requirements for a processing system. These include the size of events; the use of control or steering; the connection between events such as their ordering and stateful event processing; importance of fault tolerance; use of humans in the loop for feedback. Application processing can be characterized by the complexity of processing, the possible need for a quick turnaround, offline or online mode, closed loop or adaptive, etc. It is useful to understand the latency that can be tolerated between data collection and processing. Coupled streams, multiple streams, and interaction of streams with distributed data are important in some cases.

Examples of requirements for a processing system include, universality of interfaces; adaptive pipelines; ability to identify the important information rapidly; access control; adaptive flow control; accuracy and use of sampled data; fault tolerance and error recovery; storage matching hardware and application needs. There is a need to respond to variable rates or load changes, such as between peak and non-peak hours. This requires an elastic system capable of dynamic scalability based on the changing rates, which in turn establishes the need for programming paradigms involving unbounded data with sliding windows with the need to detect changes between windows. In most cases, the data is distributed, which a ects synchronization, parallelism, and algorithms. There is a need to characterize workflows better; for example where are computations done, are they close to the source of the data, or does data stream to the cloud?

3.2 Existing Solutions

Scientific Software Solutions:

There are a wide variety of tools and technologies that are currently used to support the integration of data streams with computational tasks in the scientific domain. Most tools and solutions however, are customized and often not easily extensible, nor interoperable with other tools.

Some scientific projects are building streaming services to handle streaming data. For example, the ATLAS experiment generates 1PB of raw detector data (DI) which is filtered to 1GB/sec of event data. A streaming service is being designed and implemented. The ability to stream data will significantly benefit the system by processing on-demand requests rather than pre-existing data. Streaming of events will allow finer grain resolution (per event) as opposed to clustering into large event collections.

Some scientific applications are exploring use of public clouds for managing and processing streaming data. For example, Ameriflux utilizes production cloud based web services including Amazon Kinesis and Lambda for collecting and processing environmental data. A challenge of such application is the creation of domain-specific functionalities while utilizing general purpose clouds, such as AWS.

In spite of impressive advances and solutions for specific project, the community however, is still missing a conceptual and comprehensive understanding of streaming systems and applications. There is a similar need and e ort for computational steering, with a specific requirement to support steering mediated by human-in-the-loop.

Given the state of the art, there is also a strong case to be made for ”building blocks” based approach to workflows, to as opposed to a single purpose general workflow system. For example, [P16-27] provides the example of EVPATH, which integrates several such reusable components. Additional specific examples of requirements include general solutions that integrate I/O and memory subsystems; streaming systems that support adequate performance without over-provisioning of hardware resources in response to problem, which is ultimately a non-scalable solution. There is a need to support unified analysis of multi-source data generation as this will become increasingly important with the emergence of sensor-based streaming data.

Industry Solutions

Streaming solutions have evolved in academic research and industry. Di erences exist between industry and research applications. One pertains to event size, which is often large in research use cases. Another di erence relates to achieving needed performance; industry might "just" add more nodes to improve performance; this is typically considered too costly for research applications. Not surprisingly these di erences have a knock-on e ect on the design and implementation of tools.

A number of streaming solutions have emerged in industry, including Apache Spark, Flink, Storm, Heron, Samza, Kinesis and MillWheel. We provide a detailed description of the two solutions — Heron and Apache Beam — that have emerged in as leading industry solutions. As they are both open source, they have potential to be very useful in new research systems.

Twitter - Heron: The analysis of user engagement, breakout moments, and many other similar use cases requires analysis of a large volume of tweets. Streaming helps in the increased collection of data and analysis in real time. As this is a system upon which applications can be built, there are no a priori constraints on the type of data that can be streamed. However, technological and implementation constraints impose practical constraints: Heron is a real-time distributed stream processing engine which uses a DAG comprising two types of nodes – called the spouts and bolts – which are the sources of stream data and stream processing centers respectively. Each spout and bolt in the topology runs multiple tasks to account for the variations in the incoming data rate and processing capabilities. The volume, variety of tweets, and the need to process hundreds of millions of events per second motivates the design and implementation of Heron.

Google - Apache Beam: Apache Beam is a unified batch and streaming data processing model which consists of data collection, processing, dataflow, and optimization. The unified system combines a stream processing system which provides continuous updates to generate an approximate real-time model with batch processing, which in turn periodically processes data to provide an exact historical model. This provides a healthy balance between correctness and latency. The Google cloud dataflow aims to provide a cloud and unified batch-stream data processing service mainly for data intensive applications.

Algorithms

Streaming data is often analyzed both "on-the-fly" and stored in an archival repository from which accumulated data can be analyzed. The latter involves traditional batch algorithms and supports many iterative machine learning approaches as data is available forever. Even for archival data, the analytics is not mature and we identify needed improvements in Section 4. However, "on-the fly" analysis o ers unique opportunities with so-called online algorithms, either looking at each data point once or just keeping those in a moving time window.

Streaming algorithms are discussed fully in Section 4.1. Some use cases discussed at the workshops are summarized below. The Astronomy use case [P16-22] [WP15-1] [P15-5] [P15-7] describes streaming principal component analysis (PCA) algorithms and the identification of "heavy hitters" (mass concentrations) corresponding to galaxy halos in astrophysical simulations. Previous work [P15-13] described parallel online clustering algorithms to find Twitter memes (information concentrations) using the Apache Storm environment [5]. Previous work [WP15-10] [P15-11] described baseline methods for electrical grid data understanding and [P16-22] stressed that in many cases systematic errors are greater than statistical errors.

Programming Models, Languages and Runtime

A number of existing models explore the problem space. For example, Fresh Breeze [WP16-4] [P16-3], a programming model and system architecture for real-time streaming applications, supports producer/consumer parallelism for streaming computations and data parallel processing for classical HPC applications. Previous work investigated application of reactive programming paradigm to distributed

stream processing [WP16-11] [P16-10], the steering of complex systems using a dynamic, data-driven modeling approach (DDDAS) with a domain specific language (DSL) and a control system for aircraft flight systems [WP16-22] [P16-26]. Streaming ideas in a compiler and runtime to manage time-dependent effects in a computer system, such as those from temperature variation which have an identified associated time window, has been investigated [WP16-5] [P16-4]. Some recent work investigates big data techniques such as MapReduce and combined them with HPC for analysis of data from simulations [WP16-1] [P16-1]; StreamMapReduce, [WP15-2] [P15-20] which allows event stream processing to work with MapReduce and extends fine-grained event driven execution models for large-data volumes; and high-performance implementation of streaming models in MPI [WP15-11] [P15-23]. Neptune [P15-16] [23] and Twitter Heron [6] address performance problems [WP15-17] in Apache Storm – in particular those associated with Apache Kafka [24]. Other industry works include Amazon [P15-1] [WP16-3], Google [P15-3,P16-11], Microsoft [P15-18] and Twitter [P16-21] [WP16-21]. It is interesting to see the transition in Google from MapReduce to Cloud DataFlow via FlumeJava and MillWheel. These are now available as open-source under Apache Beam.

4

Future Research & Development

Streaming workflows present unique challenges. The streaming use cases outlined in Section 2 cannot be fulfilled with the current software stacks running across distributed and HPC systems. Traditionally, we have a hierarchical memory (from caches to disks), a highly efficient batch queue operation that focuses on utilization of resources and a programming model that can be used to efficiently support simulation and, more recently, analyses workloads like machine learning. However, streaming operations are more interactive and can involve human in the loop. The streamed data may or may not be saved, but the online algorithm always keeps it around for a short time, perhaps just looking at it once or having a moving time window holding recent data. This fundamentally changes the current model of how scientific applications are composed and managed. For example, the traditional iterative batch algorithms must be changed as data is typically not kept across iterations. The use of files will typically only occur in saving data for later non-streaming processing. Fault-tolerance algorithms or systems that can recover essential information in real time will be required as there is typically no "rolling back".

As identified in Section 3, there are commercial, general open-source and domain-specific solutions. However, there is no consensus on the methods and technology to put this together to form modern, e ective streaming systems that can address the range of application requirements or the challenges of scale and time sensitivity that are expected in future systems. The volumes of data being generated by today's instruments and the increasing need for real-time processing require a more systematic approach to address these challenges.

In this section, we discuss some of the key research and development (R&D) topics and challenges identified during the workshop that will need to be addressed to build streaming systems. We classify future R&D in four main areas, with some challenges possibly spanning span more than one area:

- Algorithms: In Section 4.1, we identify the need for development of new algorithms to support streaming data.
- Programming models, languages, runtime and streaming software system: In Section 4.2, we identify advances needed in the spectrum of programming models, languages, runtime and associated software.
- Scientific streaming workflows: In section 4.3, we discuss the advances needed in methodologies to support streaming and human-in-the-loop in end-to-end workflows.
- Facilities: In Section 4.4, we identify the advancement in capabilities that will be needed at ASCR Facilities (HPC and networking) to address streaming and steering requirements of scientific applications.

4.1 Algorithms, Applied Mathematics and Statistics

Many stream analytics problems are complex, and the scale of unbounded streams are such that they cannot be analyzed at the rate they are produced with traditional batch algorithms. Consequently, we need di erent algorithms that can reduce the complexity of the analysis. While there is some current work in online algorithms, there is limited work tying the work to real applications. There are a number compelling research topics, including adaptive sampling, online clustering [25] and sketching.

The NRC Frontiers in Massive Data Analysis report [26] identifies seven computational giants that are essentially algorithmic challenges: Basic Statistics; Generalized N-Body Problems; Graph-Theoretic Computations; Linear Algebraic Computations; Optimizations; Integration; and Alignment Problems. The report identifies that two of the most pervasive strategies for achieving computational efficiency are sampling and parallel/distributed computing. Further, they identify four common themes:

1. State-of-the-art algorithms exist that can provide accelerations of major practical importance by significantly changing the runtime order, for example, from $O(N^2)$ to $O(NlogN)$.
2. High dimensionality in the number of variables is a persistent challenge to obtaining computational efficiency, and this demands ongoing research.
3. The non-default settings (default is sequential in memory algorithms) - streaming, disk-based, distributed, multi-threaded are quite important, yet mostly under-explored in terms of research e ort.
4. Most of the algorithms described in the NRC report [26] have only been demonstrated in research implementations. This includes, for example, the O(N logN) algorithms mentioned earlier. More work is required to create robust and reliable software before these algorithms can be used widely in practice.

The key future R&D e orts recommended for the area of algorithms include, a) developing online algorithms to support streaming towards reducing time complexity of the solutions; b) developing machine learning algorithms to support DOE applications; c) Developing parallel algorithms that can leverage multicore and future parallel computing systems.

Research Area 1: Online algorithms to support streaming towards reducing time complexity of the solutions.

Streaming and online algorithms are essential to satisfy compute/time complexity constraints of streaming applications. These algorithms are characterized by using the data only once, whereas typical iterative batch algorithms go over the data many times. Online algorithms do not require the whole dataset in advance, but they can process the input data as it arrives. There is an urgent need to extend and develop online algorithms to address the streaming requirements identified by streaming applications (Section 2).

Sampling described in Section 4.2 can go further than traditional streaming algorithms and only use a subset of the data once (or in a batch sampling many times). There are however, a number of challenges in sampling algorithms that are important to address. These include improving sampling performance; the sketch concept, which is a smaller but representative dataset; the general issues of approximation and accuracy and the trade-o between accuracy and time complexity; socially coupled systems that have intrinsic inaccuracy; where, when and what to sample; error bounds with sampling; the need for users to understand algorithms with approximations and be able to make good choices; and incremental accuracy where the initial sample results are improved as the data increases in size.

Sketch algorithms provide approximate results for queries performed over voluminous datasets. This is achieved by using data structures that serve as a surrogate (or a sketch) for the actual dataset, and then using this data structure to evaluate queries. Importantly, the space-complexity or memory footprint of this data structure is independent of the underlying voluminous dataset. Examples of such algorithms include the well-known Bloom Filter. which can be used to evaluate set memberships, described in STREAM2015

report and the Ego-net Sketching [WP16-19] [P16-19] method in STREAM2016. In essence, the space complexity (size of sketch) trade-o becomes a time-complexity versus accuracy trade-o . Frequent itemset mining over data streams[27] [28] [29], stream classification [30] and streaming clustering algorithms [31] [32] are well known, but need to be developed for massively parallel settings.

Online anomaly detection [33] [WP16-17] introduces a new robust random cut tree data structure to sketch (summarize) data. This data structure allows one to determine how anomalous a point is, by measuring the point's e ect on the size of this tree. Further, this approach is an online algorithm that can be efficiently updated when either new points arrive or old points depart. This algorithm has been made available in Kinesis [34].

Thus, there is a need to systematically evaluate existing algorithms and build on existing work to develop online algorithms to support streaming towards reducing time complexity of the solutions.

Research Area 2: Machine learning algorithms to support DOE applications.

As the volume and scale of streaming data grows, understanding data streaming will require the application of classical batch algorithms such as probabilistic models, classifiers, Bayesian, Markov, classical EnKF (Ensemble Kalman Filtering in Data Assimilation), umbrella sampling, importance sampling, Monte Carlo and other data-mining and machine learning algorithms. A number of key research questions need to be answered in this context. Is there a streaming version of every batch algorithm? When is it difficult or impossible to generate the streaming version? What are the typical performance and accuracy comparisons of streaming versus batch algorithms? How much processing should be done online and how much offline (batch)? How do the data characteristics such as volume, velocity and variety impact the appropriate algorithms for streaming? Compression algorithms and analysis techniques for compressed data are also important.

Learning methods are also critical. Manifold learning [WP16-25, P16-30] implements an advanced dimensionality reduction approach, and online methods for this are being developed. They are being applied to the detection of material morphology and structural changes in Transmission Electron Microscopy data, climate simulation visualization and clustering for metagenomics. Deep learning naturally trains (on GPU enhanced clusters) in batch but classifies in an online algorithm. Deep learning for streaming data from the National Synchrotron Light Source II at Brookhaven National Laboratory is being developed [WP16-26] [P16-31]. Reducing precision (down even to one bit) is an interesting research area for speeding up deep learning. Further, we need to research steering scenarios involving active learning: for example, a set of molecular dynamics simulations that are driven sample rare events, or machine learning classification algorithms that can take streams to improve (adaptively) their accuracy.

Information visualization and dimension reduction (e.g., [WP15-5] is critical. Dimension reduction is linear (e.g., PCA) or nonlinear (e.g., MDS) and comparison of these approaches is needed for streaming data. In previous work [WP16-7] [P16-7], streaming analysis of financial data was performed and found that dimension reduction can be applied to individual items in a time series. There remain challenges in aligning the reduced dimensions results for di erent time values, including their visualization [WP16-18][P16-18].

Quantification and reduction of uncertainty are critical to DOE applications. As streaming analysis works on incomplete data and usually only has one pass through the data, there are varying levels of uncertainty associated with the analysis results throughout the process runtime. Currently, no methods are available to sufficiently quantify this uncertainty or propose strategies to reduce it if required perhaps through the introduction of additional data sources. In particular, there is a need to provide a common basis to evaluate solutions, which in turn requires conceptual advances to determine which metrics must be evaluated.

Research Area 3: Parallel algorithms that can leverage multicore and future parallel computing systems.

In addition to developing new algorithms to handle the scale of data, there is also a critical need for scalable parallel algorithms covering multicore and parallel computing. Efficient parallelism is particularly important in some of the dimension reduction and clustering approaches discussed above with classic time consuming batch implementation which often have time complexity proportional to square of number of data points.

4.2 Programming Models, Languages, Runtime and Streaming Software Systems

Streaming systems present a number of challenges in the areas of programming models, languages, runtime and associated software. Recently, there has been extensive analysis of streaming data requirements in the research community, but typically not with an architecture and system optimized for the streaming aspects of the problem. Programming models and software solutions have evolved in the commercial space of big data systems. However, their applicability to science applications is still unclear.

Today, programming models for streaming are still exemplified by point solutions and not abstracted. *The programming and runtime requirements of streaming applications are di erent from the traditional simulations. There is an urgent need to evaluate existing commercial programming models and develop new abstractions that meet the needs of DOE streaming applications.*

Science use cases exhibit complex workflow management problems, where data moves from tasks to task and often involve a heterogeneous set of resources and research specialists. One finds data in motion both when generated by instruments and also between steps of complex workflows. The latter often transfer file objects that can be packetized to look like events, which are naturally larger than those seen in instrument data processing. Further those scientific data events are typically larger than those found in commercial applications where an event is like a Tweet or a web transaction. It is necessary to understand the di erences that impact the needed streaming system and programming model. It is important to understand if there are lessons learned from complex event processing that can be applied to next- generation streaming system design.

Thus, it is critical to evaluate existing and research new streaming programming models that address hardware and application challenges. *We identify three research areas that need to be explored further: a) We need to evaluate existing programming models on functionality, performance and usability for streaming applications; b) We need to explore the impact of emerging technologies such as new memory technologies and new storage ideas such as object stories; c) We need to develop streaming programming models to address performance and programmability.*

Research Area 1: Programming models for streaming applications.

Current practice in programming models and runtime can be divided into three types: commercial solutions, open source (Apache) and domain-specific solutions. These di erent areas lead to tremendous diversity and rapid evolution in stream analytics programming models and systems. As there is no common understanding of application requirements and programming models, making them impractical to compare authoritatively.

Appropriate programming abstractions that address the needs of streaming applications and implementations that lower the programming burden need to be developed. To do so, there is a need to evaluate existing programming models and identify gaps and challenges in their use for streaming applications. There is a need to identify best practices and computational paradigms that can raise the level of abstraction to simplify the programming tasks for end users, removing the requirements of expertise in distributed computing, ad hoc analytics and integrating diverse software packages.

There are a number of existing industry solutins (e.g., Apache Beam, Storm, Heron) systems that need to be evaluated in the context of DOE applications, and approaches should be compared to existing solutions to both workflow and streaming Further, Domain-specific languages are a popular feature in many parallel computing areas. They do not seem to have been explored deeply for streaming systems.

Research Area 2: Hardware and software architecture on distributed and HPC systems to support stream and dataflow scenarios.

Streaming will always give rise to data movement, which needs to be efficiently integrated with processing. Traditional science computing models are based on batch executables, but that is not appropriate for always-on event streams. There is a need for parallel dataflow capabilities, but that has been hard to express and implement as there is no consensus on data placement and programming models.

There is a need for systematic rethinking of the hardware and software architecture to support stream and dataflow scenarios. We need to study the impact of disruptive innovations (such as new memory technology) on the entire streaming software stack including programming models. Additionally, there is a need to understand if technologies such as distributed object stores are a useful abstraction for streaming.

Event and stream queries need novel data management and computing infrastructure that matches the application requirements, such as application-driven choices for what and how much data to store and how to store it. There is also a need to dynamically convert streams into scientific and analytical data types within workflows and make automated processing of such data possible at the rates that the data is being produced.

This requires advances in efficient parallel data management and evaluation of the abstractions provided by Spark RDD [3], AsterixDB [35] and Tachyon [36]. There are a number of research questions in this space, including: Can ideas from adaptive meshing be used to handle key partitioning and sharding? Is MPI I/O and/or lessons from it useful? How can the community move away from file-based solutions and develop the ability to dynamically process streams?

In the hardware, we need a rethinking of the balance between bandwidth, storage and compute. Further use of modern network hardware to support reservations and science DMZs will also be important (more discussion in Facilities Section 4.4). It is also necessary to evaluate the use of software designed for exascale runtimes in streaming systems. For example, there is a need to support of dynamic inhomogeneous threads that could be very useful in streaming.

Research Area 3: Programming abstractions that address performance and programmability.

Performance is a critical issue both for streaming and batch systems. Recent work [37] has compared reduction operations in MPI, Spark and Flink. The dataflow model of the Apache systems is attractive but leads to performance degradation. What are the (acceptable) costs of layers of abstraction and what is the relationship of programmability and performance? Practical programmers tend to think in terms of conventional abstractions, which can be complex to map to streaming and dataflow concepts. However, we should explore building streaming abstractions on top of familiar batch ideas as this approach could open up the field to a broader range of users.

There is a need for programming abstractions and runtime support for adaptive and steered applications in large-scale simulation science. Programming abstractions need the ability to represent and track time, that is missing in today's batch processing, but is important in stream processing.

The performance of two big data languages – Python and Java [37] – could be significantly improved if their compilers and runtime were approached in a way that is well understood for simulation languages C++ and Fortran. More understanding is needed of the Google concepts of bounded or unbounded and the relationship to batch and streaming programming [P15-3].

An interesting abstraction is map streaming [38] which is a high- level description of systems like Apache Storm that describes parts of the system but not all; for example, Storm supports pub-sub and dataflow

but not the full parallel computation useful in some applications such as SLAM [WP15-17]. The relation of these streaming concepts and abstractions to workflows needs further study.

Similarly, there are other areas that provide potentially interesting and relevant approaches, e.g., SQL query optimization [WP15-16] hasn't been fully exploited for distributed systems, although this has been studied in the context of data streams [39] [40]; the Apache Flink project with its dataflow architecture is looking into this [4]. Actor models – natural for events – should also be studied.

Scheduling processing streams on HPC systems is difficult compared to traditional batch jobs. Optimal stream scheduling where you also try to maximize a performance measure such as latency or throughput is NP-hard. There are, however, possible stochastic solutions to the problem and heuristics based on previous runs can be used to efficiently find good approximate solutions [41]. Further research on this could be of benefit to many streaming applications. There is also the trade-o between latency and throughput in stream processing settings. Further research is needed to better understand this and other stream optimizations [42]. In particular, there is a need to develop methods for scheduling and placement of processing and its relations to in situ solutions for HPC and the cloud-fog-device model for IOT use cases.

4.3 Scientific Streaming Workflows

Streaming workflows are increasingly becoming critical components of the scientific software ecosystem since it provides a convenient way to represent and orchestrate distributed and high performance computation and related data dependencies. Workflows can be used to capture and orchestrate the distributed set of resources, data and collaborators that might be involved in the end-to-end streaming application. *There is a critical need and a gap in current methodologies and tools that enable human-in-the-loop and interactivity in scientific workflows.*

Human-in-the-loop is a requirement across many applications including light sources, materials design and analysis. For example, scientists using light sources such as the Advanced Light Source (ALS) or Advanced Photon Source (APS) often would use coarse-grained analyses during the course of the experiment, with the human-in-the-loop that is used to make decisions about experiment setup.

The NRC Frontiers in Massive Data Analysis report [26] has also addressed "Human Interaction with Data" and mentions a number of topics including data visualization and exploration, crowdsourcing and hybrid human/computer data analysis. Tools such as IPython [43]/Jupyter [44] and pandas [45] provide interactive data analysis capabilities. However, interactive data capabilities on streams of data at supercomputing scale processing is still relatively unexplored.

Previous DOE ASCR workshops, "The Future of Scientific Workflows" [46] and "Management, Analysis, and Visualization of Experimental and Observational Data" [47] have looked at a broader scope of workflows from the simulation and for experimental and observational data. In this section, we highlight the capabilities specifically needed to support streaming and steering.

We identify three core research areas in scientific workflows: a) developing a classification of streaming application to understand the needs; b) developing QoS metrics for end-to-end data streaming pipelines; c) automated, semi-automated and manual steering.

Research Area 1: Classification of streaming applications.

We need a thorough understanding of the end-to-end streaming workflows to understand the complexity and breadth of requirements. The classification can influence research and development in a number of key areas including programming models, system software and development of benchmarks for evaluating existing and new solutions.

Streaming applications can be classified by various dimensions. Size of data and rate of data are often considered. Streaming problems might also be classified by the processing delay tolerated: overnight to plan future experiments (e.g., telescopes, day-shift experiments); hourly to detect problems (e.g., stable, long-term HEP experiments) or minutes to maintain steady-state data follow experiment evolution (e.g., time-resolved, in situ experiments).

There is a need to engage with DOE facilities and projects using human computer interaction (HCI) methodologies to build a deeper understanding of the usage models and gaps in the end-to-end workflow characteristics. Understanding how scientists interact with streams and use steering can inform where existing software might be used and identify gaps that are specific to scientific needs. The classification can also facilitate other activities, such as modeling and development of benchmarks and mini-apps.

Research Area 2: QoS metrics for end-to-end data streaming pipelines.

Streaming data is tied to time-critical experiments where success of the experiments depends on the reliable performance of the overall system. As scientists deal with the instruments, they need workflow tools that can help them meet the performance guarantees while being easy to use and manage.

It will be necessary to determine the right QoS metrics that are applicable for streaming and steering applications and to develop capabilities in tools that let users orchestrate their complex flows while meeting the needs of the users. We need to develop advanced capabilities in tools and libraries to allow for composition and execution of streaming data and processing.

Streaming data and processing requires specific workflow capabilities. For example, streaming data can often be a multi-source or multi-subscriber data analysis infrastructure. This needs the ability to handle geographically distributed data streams and resources e ectively and synchronize with the analysis pipeline. The middleware for streaming systems is important and needs to efficiently move data in and out of di erent services running on potentially di erent enclaves.

E ective interaction of humans and computers is key in all the systems, but in particular in those with increased autonomy, such as streaming systems. The researcher needs to be able to e ectively express goals and constraints that need to be enforced by the system. The system needs to communicate when it is unable to meet the original goals set by the user. Similarly, streaming needs abstractions for data-at-rest and data-in-motion that are not widely considered in scientific computing focused on high performance computing. Similarly, we need visualization capabilities to allow users to interact with the resulting data. Capturing provenance through the end-to-end workflow is also important.

Research Area 3: Automated, semi-automated and manual steering.

Steering is an essential capability that is increasingly needed at multiple levels and parts of the systems. We need to develop capabilities for enabling steering at the user level and system level. It might be necessary to steer the simulations, the sensors or experiments or to steer the data stream itself. Additionally, it is important to understand and evaluate the thresholds and timescales at which human and automated intervention might be practical and necessary. For example, scientists might prefer manual intervention when handling expensive instrument time and may not wish to rely entirely on a machine learning algorithm. It is necessary to build the capabilities to allow controls for both scenarios and the transition between the human and the machine.

4.4 Facilities

Given the volumes and rates of streamed data, experimental and observational facilities will require the use of HPC/leadership computing capabilities. This will require substantial R&D into the software systems and capabilities that these systems provide. There are at least two important drivers of change: Historically, HPC/leadership computing facilities have been net producers of data. With HPC resources becoming

datastream endpoints they will become equal producers as well as consumers of data. Second, HPC systems and their software have been mostly designed to support the efficient execution of monolithic applications to be executed in essentially batch mode. The batch mode of execution is fundamentally at odds with the requirements of streaming applications. Furthermore, human-in-the-loop and real-time constraints will stress the capabilities of networks [P16-9], as will its integration with core computational facilities. *The needs of streaming are important and require invasive changes to the software and middleware stack deployed on HPC/leadership computing resources.*

Several opportunities spanning research, development and operational aspects will need to be considered for next-generation streaming systems supported through DOE computing facilities [P16-28]. Specifically, the proposed R&D includes: *a) Develop abstractions and solutions that integrate stream and batch-queue systems. b) Policies and technologies for co-scheduling of heterogeneous distributed components. c) Infrastructure support for online and machine learning algorithms needed by streaming applications. d) Investigate and deploy security mechanisms suitable for streaming data [P16-15][WP16-15].*

Many of the challenges in this section have similar roots as those in earlier sections (such as algorithms), but the nature of the solutions are more to do with deployment and production, as opposed to core research.

Research Area 1: Abstractions and solutions that integrate stream and batch-queue systems

HPC facilities and ESnet will need to consider the impact of streaming applications on the abstractions and solutions provided at the center. Careful consideration of the following research questions will be needed. What changes will be required to the (i) architecture and design, and (ii) middleware and software environment of leadership class and other facilities if they are to support streaming science workloads e ectively? What is the role of deep- memory hierarchies and advances in data storage? How can existing abstractions for fine-grained resource management in HPC systems (such as the pilot-job abstraction) be extended, both in concept and as software implementations, to support the ingest of real-time data without a fundamental re-evaluation of the existing landscape?

As highlighted earlier, there is a plethora of streaming systems that provide local solutions that are not adequately general or extensible. As performance and functional requirements of streaming systems increase, the complexity and challenges of providing a versatile software ecosystem will increase. This requires design studies leading to conceptual frameworks that support comparative analysis of functionality and performance of streaming systems in the production settings in which they will ultimately operate.

Such a powerful software environment will only be successfully designed if we have a good collection of mini-apps and benchmarks for streaming and steering applications.

Research Area 2: Policies and technologies for co-scheduling of heterogeneous distributed components.

Streaming applications bring the need for concurrent utilization of resource types that have semantically and syntactically heterogeneous interfaces, di erent measures of successful utilization, policies and different administration organizations. There is a need to evaluate how they can be provisioned so as to support the requirements of the distributed workflows arising from streaming and steered applications. How can compute resources be scheduled in anticipation of experiments and observational data becoming available?

Current operational objectives promote total resource utilization over all other metrics. Supporting and responding to real-time workloads will require resource management approaches that are not tuned for maximizing resource utilization but prioritize user- and application-level metrics.

There is a need to examine the policies and operational objectives of HPC to support streaming and steering applications. For example, in order to support the burstiness of the applications that involve streaming data, policies related to the number and duration of jobs, as well as queues that support a wider range

of priority definitions, are needed. What changes will be required to the scheduling policies if they are to support streaming science workloads e ectively? What kind of allocation and charging policies are suitable to accommodate real-time jobs? How can network hardware and architecture support reservations and science DMZs? [P16-8][WP16-8].

Research Area 3: Infrastructure support for online algorithms and machine learning for data reduction.

The facilities of the future must provide the basic infrastructure such that the velocity of processing must be commensurate with that of data generation. Ingesting data rates of 100GB/s successfully into the computational workflow is going to be a major challenge on the middleware and workflow systems. Facilities will need to provide appropriate methods to access hardware advances (e.g., burst bu er [48]) that can help manage these challenges.

Where the applications and middleware cannot successfully manage data rates and volumes, there is a critical need for middleware services that are built upon research advances in online machine-learning algorithms and approaches for data reduction (as discussed in Section 4.1). These production services should become a critical component of the infrastructure.

5

Community Activities

Participants at both STREAM2015 and STREAM2016 identified that the streaming community is still nascent. It was recognized that, in addition to the R&D e orts, community building activities are essential to create and grow the research areas in streaming. *Community building activities that include engagement across scientific domains, academia and industry are needed.*

The workshop participants identified key areas of recommendation for community-building activities across various federal agencies:

- Building and growing the community through meetings, a clearinghouse of community information and activities.
- Build a community consensus around application benchmarks to evaluate existing and new solutions for streaming and steering.
- Develop streaming software libraries that enable streaming solutions to be widely adopted.
- Build community-wide education and training e orts towards workforce development.

We discuss these recommendations in detail in the sections below.

5.1 Building and growing the community

E orts need to be made to build and grow the community around streaming that spans government labs, academia and industry. Workshop attendees identified a need for a clearinghouse of community information and activities, and it was suggested that the STREAM workshop website [49] could serve as this clearinghouse.

Additional e orts will be required to organize community activities, including application surveys, special issues, workshops and industry engagement through advisory committees. There is a need to identify and form multiple subgroups covering topics identified in the two STREAM meetings and reports. The community would benefit from a survey of existing software systems (highlighting those in open source) and a survey of applications (existing and potential). Another useful consideration would be to consider streaming data challenges following either the successful provenance challenge [50] or those popularized by Kaggle [51]. Streaming software developers should consider working with foundations such as Apache to improve support of scientific applications [52].

While there are definitions of streaming and steering, there is a recognition that this terminology needs to be refined and will evolve with time as the community around the topic matures.

Several existing conference and meeting venues were discussed to grow the community. A BoF or workshop at the Supercomputing conference was seen as promising. Journal special issues and edited books should also be explored by the community.

E orts to engage additional key people and activities that could not make it to the workshops should continue. This includes industry representatives from GE, IBM, Facebook and the financial sector. Strategic partnerships between academia and industry will be key to enabling next-generation streaming. Summer internships can pave the way for engaging students, faculty and sta , and strategic partnerships and programs that fund projects across industry and academia will help move the field.

5.2 Benchmarks and Application Collections and Scenarios

We require an understanding of the true complexity and breadth of application requirements to make substantive progress on algorithms, programming models and software systems for streaming data analytics and steering. Similar to grand challenges in domain sciences, the development of a series of representative streaming and steering examples will be of enormous value. These should include representative data sets at scale. Linear road is an early benchmark developed around the Aurora streaming system focusing on one application [53]. There may be a need for benchmark suites like TPC or HPCC that foster collaboration between industry and the research community, but we need benchmarks that cover the complexity of the distributed streaming area. There is a clear need to investigate the performance of streaming applications and software on clouds and HPC systems. This need is covered in this subsection for applications and in the next subsection for software.

Initial candidate benchmarks were considered at STREAM2015, e.g., [WP15-2] [P15-20] (taxicab data), [WP15-11] [P15-23] (based on STREAM memory benchmark), [WP15-16] (LinkedIn monitoring data) and [WP15-17] [P15-19] (SLAM robotic planning). Many of the other STREAM2015 talks implicitly defined possible benchmarks, e.g., [WP15-1] [P15-5] [P15-7] with streaming PCA, [P15-13] with online clustering algorithms and [WP15-10] [P15-11] with smart electrical grids. Zheng has compiled several interesting open datasets for urban studies [54]. Other examples were given at STREAM2015, including ATLAS-LHC analysis, avionics, drug discovery and Galaxy Zoo [55]. The 2016 SPIDAL report presents many open applications with data analytics that could be converted to benchmarks [52]. The rich application discussion in Section 2 of the STREAM2016 report is particularly relevant. The inspiration for many of the benchmarks must come from industry and must involve interesting unrestricted datasets. Further, we need to include benchmarks that present real-time data from real-time sources. A simple initial task is to agree on a Hello Streaming World example. Many recent data science classes include lists of available open datasets; example, an Indiana big data class involved 45 student projects with 39 datasets and 91 technologies[56].

We need a more comprehensive understanding and classification of streaming applications to scope a benchmark set. An initial classification was provided in STREAM2015 report[1]. This could build on the Dwarfs [57] for parallel computing and the Ogres [58] [59] for big data applications. Some of the streaming characteristics include the type of events, the size of events and the lifetime of events.

Validation and verification implies a need for providing test environments in which analysis algorithms can be evaluated against existing solutions in terms of speed, accuracy and added value. Such a test range would be useful to demonstrate and quantify the impact of di erent data streams on the accuracy and value of the analysis result for the subsequent decision-making process. In addition, however, there is a need to validate the analytical and decision models at runtime to determine whether they are still correct or if they need adjustment?

There is a need for number N (e.g., between 6 and 20) of exemplar problems that identify use cases clearly and are well described. From this collection, challenges, abstractions, scaling, theory, stream operators, algorithms, languages and tools should be extracted. The use cases can then be implemented with current software on current hardware and what works and what doesn't can be identified. This process will generate a robust benchmark set. Note some examples can involve actual streaming data and others a repository of data that can be used to generate streams.

5.3 Streaming Software System and Algorithm Library

There is a need to assemble a general-purpose, high-utility, open source scientific streaming software library that can be used by the community. It should be carefully integrated with the Apache Foundation software. It would hold enhancements to Apache software (for example, Apache Storm and Mahout [5]) as well as standalone software. It would contain systems (middleware) and algorithms and would be streaming analog to SPIDAL and MIDAS [60] [52]. It would need to have broadly applicable functions, support data reuse, and implement a suite of algorithms exploiting key streaming ideas, such data sketches and certainly the challenges highlighted in NRC's computational giants. It would also need to have core capabilities, such as state-of-the-art tools to allow Java and Python to run at high performance.

We need to aim for a toolkit that will support the stringent requirements of the major streaming science applications but that is architected so it can be used across domains, including LHC analysis, light sources, astronomy and simulation data. The development of a community toolkit should not occur before the primary set of requirements are in place, research has been conducted and system support is understood and has stabilized.

5.4 Education and Training

Streaming data applications are seen as critical to economic growth and development, and they present a tremendous opportunity for workforce development. In addition, a growing number of scientific and engineering disciplines are being inundated with streaming data, either from observational facilities or through simulations. As a result, an understanding of the emerging tools and environments for streaming data will become increasingly important for future skilled workers. Understanding the best practices and developing curriculum support was universally regarded as essential at the workshop. Industry participants agreed there is tremendous, and growing, demand for these skills and were supportive of exploring internships and partnership programs.

The stream workshop talks highlighted a number of current and future education and training e orts and needs. Brunner [P15-8] stressed the relationship between a streaming systems curriculum and the broader area of data science and discussed an interesting link to a data science incubator at UIUC. Braverman at John Hopkins, has already integrated a discussion on new algorithms needed for streaming data into courses [61]. The workshop participants widely agreed that data science education, and in particular education in streaming data applications, requires realistic datasets—in particular from industry.

6

Summary

This workshop report summarizes the discussions, findings and recommendations from STREAM 2016: Streaming Requirements, Experience, Applications and Middleware Workshop; held in Tysons, Virginia in March 2016. The workshop surveyed a large number of applications from the Department of Energy Office of Science as well as others identifying requirements, gaps and challenges.

The workshop discussions identified future R&D in four key areas Algorithms; Programming Models, Languages and Runtime Systems; Human-in-the-loop and Steering in Scientific Workflow; and Facilities. There was also a recognition for a need to build a community and associated activities around streaming and steering for science applications across the agencies.

Appendix A: References

A.1: Bibliography

[1] G. C. Fox, L. Ramakrishnan, and S. Jha, "Stream2015 final report," 2015. [Online]. Available: https://doi.org/10.13140/RG.2.1.3907.2240

[2] "Spark." [Online]. Available: http://spark.apache.org/

[3] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauly, M. J. Franklin, S. Shenker, and I. Stoica, "Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing," in *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. San Jose, CA: USENIX, 2012, pp. 15–28. [Online]. Available: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/zaharia

[4] A. F. Batch and S. D. P. System. [Online]. Available: https://flink.apache.org/

[5] A. S. S. Software. [Online]. Available: http://storm.apache.org/

[6] S. Kulkarni, N. Bhagat, M. Fu, V. Kedigehalli, C. Kellogg, S. Mittal, J. M. Patel, K. Ramasamy, and S. Taneja, "Twitter heron: Stream processing at scale," in *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*. ACM, 2015, pp. 239–250.

[7] A. S. S. Software. [Online]. Available: http://samza.apache.org/

[8] "Kinesis." [Online]. Available: https://aws.amazon.com/kinesis/

[9] T. Akidau, A. Balikov, K. Bekiroglu, S. Chernyak, J. Haberman, R. Lax, S. McVeety, D. Mills, P. Nordstrom, and S. Whittle, "Millwheel: Fault-tolerant stream processing at internet scale," in *Very Large Data Bases*, 2013, pp. 734–746.

[10] "ATLAS." [Online]. Available: https://home.cern/about/experiments/atlas

[11] P. Calafiura, K. De, W. Guan, T. Maeno, P. Nilsson, D. Oleynik, S. Panitkin, V. Tsulaia, P. V. Gemmeren, and T. Wenaus, "The atlas event service: A new approach to event processing," *Journal of Physics: Conference Series*, vol. 664, no. 6, p. 062065, 2015. [Online]. Available: http://stacks.iop.org/1742-6596/664/i=6/a=062065

[12] P. Calafiura, K. De, W. Guan, T. Maeno, P. Nilsson, D. Oleynik, S. Panitkin, V. Tsulaia, P. Van Gemmeren, and T. Wenaus, "Fine grained event processing on hpcs with the atlas yoda system," in *Journal of Physics: Conference Series*, vol. 664, no. 9. IOP Publishing, 2015, p. 092025.

[13] "Belle II." [Online]. Available: https://www.belle2.org/

[14] F. An *et al.*, ""a side-by-side comparison of daya bay antineutrino detectors "," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 685, pp. 78 – 97, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016890021200530X

[15] J. Deslippe, A. Essiari, S. J. Patton, T. Samak, C. E. Tull, A. Hexemer, D. Kumar, D. Parkinson, and P. Stewart, "Workflow management for real-time analysis of lightsource experiments," in *Proceedings of the 9th Workshop on Workflows in Support of Large-Scale Science*. IEEE Press, 2014, pp. 31–40.

[16] J. Blair, R. S. Canon, J. Deslippe, A. Essiari, A. Hexemer, A. A. MacDowell, D. Y. Parkinson, S. J. Patton, L. Ramakrishnan, N. Tamura, B. L. Tierney, and C. E. Tull, "High performance data management and analysis for tomography," pp. 92 121G–92 121G–9, 2014. [Online]. Available: http://dx.doi.org/10.1117/12.2069862

[17] B. J. Quiter, M. S. Bandstra, T. H. Joshi, J. Maltz, A. Zoglauer, and K. Vetter, "Characterization of an advanced airborne radiation detector system for the ares project," in *Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2015 IEEE*. IEEE, 2015, pp. 1–3.

[18] M. S. Bandstra, T. J. Aucott, E. Brubaker, D. H. Chivers, R. J. Cooper, J. C. Curtis, J. R. Davis, T. H. Joshi, J. Kua, R. Meyer *et al.*, "Radmap: The radiological multi-sensor analysis platform," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 840, pp. 59–68, 2016.

[19] P. Northwest. National Laboratory "Analysis in Motion" Initiative. [Online]. Available: http://aim.pnnl.gov

[20] "The earth system grid federation." [Online]. Available: http://esgf.llnl.gov

[21] "Esnet esmond." [Online]. Available: http://software.es.net/esmond/

[22] "ESxSNMP." [Online]. Available: https://code.google.com/archive/p/esxsnmp/

[23] T. Buddhika and S. Pallickara, *Neptune: Real Time Stream Processing for Internet of Things and Sensing Environments. (To appear) Proceedings of the 30th IEEE International Parallel & Distributed Processing Symposium*. USA: Chicago, 2016.

[24] A. K. P.-S. Environment. [Online]. Available: http://kafka.apache.org/

[25] S. Guha, A. Meyerson, N. Mishra, R. Motwani, and L. O'Callaghan, "Clustering data streams: Theory and practice," *IEEE Trans*, vol. 15, no. 3, pp. 515–528, March 2003.

[26] N. Research, *Council "Frontiers in Massive Data Analysis"*. Washington, DC: The National Academies Press, 2013.

[27] Y. Chi, H. Wang, P. S. Yu, and R. R. Muntz, "Moment: Maintaining closed frequent itemsets over a stream sliding window," *In*, vol. 2004, pp. 59–66, 2004.

[28] R. Jin and G. Agrawal, "An algorithm for in-core frequent itemset mining on streaming data," *In*, vol. 8, 2005.

[29] J. H. Chang and W. S. Lee, "Finding recent frequent itemsets adaptively over online data streams," in *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2003, pp. 487–492.

[30] P. Domingos and G. Hulten, "Mining high-speed data streams," in *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2000, pp. 71–80.

[31] L. O'callaghan, A. Meyerson, R. Motwani, N. Mishra, and S. Guha, *Streaming-data algorithms for high-quality clustering*. p. 0685. IEEE: In *icde*, 2002.

[32] M. Charikar, L. O'Callaghan, and R. Panigrahy, "Better streaming algorithms for clustering problems," in *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*. ACM, 2003, pp. 30–39.

[33] S. Guha and O. Schrijvers, "Robust random cut forest based anomaly detection on streams," in *Proceedings of The 33rd International Conference on Machine Learning*, 2016, pp. 2712–2721.

[34] "Random cut forest algorithm in kinesis." [Online]. Available: http://docs.aws.amazon.com/kinesisanalytics/latest/sqlref/random-cut-forest.html

[35] S. Alsubaiee, Y. Altowim, H. Altwaijry, A. Behm, V. Borkar, Y. Bu, M. Carey, I. Cetindil, M. Cheelangi, K. Faraaz *et al.*, "Asterixdb: A scalable, open source bdms," *Proceedings of the VLDB Endowment*, vol. 7, no. 14, pp. 1905–1916, 2014.

[36] H. Li, A. Ghodsi, M. Zaharia, S. Shenker, and I. Stoica, "Tachyon: Reliable, memory speed storage for cluster computing frameworks," in *Proceedings of the ACM Symposium on Cloud Computing*. ACM, 2014, pp. 1–15.

[37] S. Ekanayake, S. Kamburugamuve, P. Wickramasinghe, and G. C. Fox, "Java thread and process performance for parallel machine learning on multicore hpc clusters," August 8, Technical Report, 2016. [Online]. Available: http://dsc.soic.indiana.edu/publications/ieee.bigdata.2016-v9.pdf

[38] G. C. Fox, S. Jha, J. Qiu, and A. Luckow, *Towards an Understanding of Facets and Exemplars of Big Data Applications*. in 20 Years of Beowulf: Workshop to Honor Thomas Sterling's 65th Birthday. October 13-14Annapolis, 2014.

[39] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom, "Models and issues in data stream systems," in *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*. ACM, 2002, pp. 1–16.

[40] A. Arasu, B. Babcock, S. Babu, J. Cieslewicz, M. Datar, K. Ito, R. Motwani, U. Srivastava, and J. Widom, *Stream: The Stanford data stream management system*. Book chapter, 2004.

[41] D. Carney, "Uur etintemel," in *Operator scheduling in a data stream manager*. Stan Zdonik, Mitch Cherniack, and Mike Stonebraker. In *Proceedings of the 29th international conference on Very large data bases-Volume 29*, . VLDB Endowment: Alex Rasin, 2003, pp. 838–849.

[42] M. Hirzel, R. Soul, S. Schneider, B. Gedik, and R. Grimm, "A catalog of stream processing optimizations," *ACM Comput. Surv.*, vol. 46, p. 4, March 2014.

[43] F. Pérez and B. E. Granger, "IPython: a system for interactive scientific computing," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 21–29, 2007.

[44] "Jupyter." [Online]. Available: http://jupyter.org

[45] W. McKinney, "pandas: a foundational python library for data analysis and statistics," *Python for High Performance and Scientific Computing*, pp. 1–9, 2011.

[46] "The Future of Scientific Workflows: Report of the DOE NGNS/CS Scientific Workflows Workshop," 2015.

[47] E. W. Bethel and M. G. (eds.), "Report of the DOE Workshop on Management, Analysis, and Visualization of Experimental and Observational data – The Convergence of Data and Computing," Lawrence Berkeley National Laboratory, Berkeley, CA, USA, 94720, Tech. Rep., May 2016, lBNL-1005155.

[48] N. Liu, J. Cope, P. Carns, C. Carothers, R. Ross, G. Grider, A. Crume, and C. Maltzahn, "On the role of burst bu ers in leadership-class storage systems," in *012 IEEE 28th Symposium on Mass Storage Systems and Technologies (MSST)*. IEEE, 2012, pp. 1–11.

[49] "Stream website." [Online]. Available: http://streamingsystems.org/

[50] L. e. a. Moreau, "Special issue: The first provenance challenge," *Concurrency Computat.: Pract. Exper.*, vol. 20, p. 409, 2008.

[51] "Kaggle." [Online]. Available: https://www.kaggle.com/

[52] I. U. (Fox, C. Qiu, R. J. von Laszewski, and V. T. (Marathe), "Kansas (paden)," *Stony Brook (Wang), Arizona State (Beckstein), Utah (Cheatham) "Datanet: CIF*, vol. 21, pp. 14–43 054, July 2016. [Online]. Available: http://dsc.soic.indiana.edu/publications/SPIDAL-DIBBSreport_July2016.pdf

[53] A. Arasu, M. Cherniack, E. Galvez, D. Maier, A. S. Maskey, E. Ryvkina, M. Stonebraker, and R. Tibbetts, "Linear road: a stream data management benchmark," in *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 2004, pp. 480–491.

[54] Y. Zheng, *Urban Computing at Microsoft has talks*. open datasets and codes for urban planning and other aspects where computing can help urban life. [Online]. Available: https://www.microsoft.com/en-us/research/project/urban-computing/

[55] C. Science. Galaxy classification project. [Online]. Available: http://www.galaxyzoo.org/

[56] "Bdaa fall 2015." [Online]. Available: http://dsc.soic.indiana.edu/publications/List

[57] K. Asanovic, R. Bodik, B. C. Catanzaro, J. J. Gebis, P. Husbands, K. Keutzer, D. A. Patterson, W. L. Plishker, J. Shalf, S. W. Williams, and K. A. Yelick, "The landscape of parallel computing research: A view from berkeley," *December*, vol. 18, 2006. [Online]. Available: http://www.eecs.berkeley.edu/Pubs/TechRpts/2006/EECS-2006-183.html

[58] G. C. Fox, S. Jha, J. Qiu, S. Ekanayake, and A. Luckow, "Towards a comprehensive set of big data benchmarks," in *Chapter in Big Data and High Performance Computing*, L. Grandinetti and G. Joubert, Eds. Editors IOS, 2015.

[59] G. Fox, J. Qiu, S. Jha, S. Ekanayake, and S. Kamburugamuve, "Big data, simulations and hpc convergence," January 30 2016.at WBDB 2015 Seventh Workshop on Big Data Benchmarking, New Delhi India, December 14, Technical Report, 2015. [Online]. Available: http://dsc.soic.indiana.edu/publications/HPCBigDataConvergence.pdf

[60] "SPIDAL." [Online]. Available: http://spidal.org/

[61] "Vladimir Braverman's website." [Online]. Available: http://www.cs.jhu.edu/~vova/

A.2 STREAM2016 References

STREAM2016 Workshop Presentations

P16-1 Gagan Agrawal. Can Commercial BigData Ideas Benefit Analysis of Instrument Data?

P16-2 Mohsen Amini. HLSAAS: HIGH-LEVEL LIVE VIDEO STREAMING AS A SERVICE

P16-3 Jack B. Dennis. Programming Model and Architecture for Real Time Streaming P16-4 Chen Ding. Timescale Stream Statistics for Hierarchical Management

P16-5 Salman Habib. Streaming Data in Cosmology

P16-6 Marty Humphrey. Leveraging Public Clouds for DOE Environmental Streaming Data

P16-7 Geoffrey Fox, Supun Kamburugamuve. WebPlotViz: Browser Visualization of High Dimensional Streaming Data with HTML5

P16-8 Dimitrios Katramatos. Streaming Data Analysis on the Wire

P16-9 Raj Kettimuthu. Computing and networking challenges in supporting streaming applications

P16-10 Shweta Prabhat Khare. Distributed Reactive Stream Processing

P16-11 Eugene Kirpichov. Dataflow / Apache Beam - A Unified Model for Batch and Streaming Data Processing

P16-12 Scott Klasky. Stream Processing for Remote Collaborative Data Analysis

P16-13 Kerstin Kleese Van Dam. Reliable Performance for Streaming Analysis Workflows

P16-14 Andre Luckow. Pilot-Streaming: Design Considerations for a HPC Stream Processing Framework

P16-15 Andre Martin. Elastic and Secure Energy Forecasting in Cloud Environments

P16-16 Benji Maruyama. Autonomous Experimentation Applied to Carbon Nanotube Synthesis

P16-17 Nina Mishra. Robust Random Cut Forest Based Anomaly Detection On Streams

P16-18 Klaus Mueller. Mining Behavior Patterns in Streaming Multivariate Data

P16-19 Srinivasan Parthasarathy. Ego-net Sketching for Streaming Graph Analytics

P16-20 Brian Quiter. Radiological Search – a Long-Standing Streaming Application

P16-21 Karthik Ramasamy. Streaming in Practice

P16-22 Alex Szalay. Streaming in Astronomy

P16-23 Nathan Tallent. Processing Streaming Data In High Energy Physics Workflows

P16-24 Craig Tull. Real-time Streaming Analysis for BES User Facilities

P16-25 Vakhtang Tsulaia. Streaming in ATLAS

P16-26 Carlos Varela & Stacy Patterson. Steering Complex Systems using a Dynamic, Data-Driven Modeling Approach

P16-27 Matt Wolf. Rethinking streaming system construction for next-generation collaborative science

P16-28 John Wu. Technology for Distributed Streaming Analytics

P16-29 Dean Williams. The Earth System Grid Federation (ESGF)

P16-30 Shinjae Yoo. Streaming Manifold Learning and DOE Applications

P16-31 Dantong Yu. Deep learning for analyzing NSLS-II data stream

STREAM2016 Workshop White Papers

WP16-1 Gagan Agrawal, Can Commercial BigData Ideas Benefit Analysis of Instrument Data?

WP16-2 Mohsen Amini, HLSAAS: HIGH-LEVEL LIVE VIDEO STREAMING AS A SERVICE

WP16-3 Roger Barga, Moving Towards Streaming-Data Analysis

WP16-4 Jack B. Dennis, Programming Model and Architecture for Real Time Streaming

WP16-5 Chen Ding, Timescale Stream Statistics for Hierarchical Management

WP16-6 Marty Humphrey, Leveraging Public Clouds for DOE Environmental Streaming Data

WP16-7 Geoffrey Fox, Supun Kamburugamuve, WebPlotViz: Browser Visualization of High Dimensional Streaming Data with HTML5

WP16-8 Dimitrios Katramatos, Streaming Data Analysis on the Wire

WP16-9 Darren J. Kerbyson, Processing Large Scale Streaming Data from High Energy Physics Workflows

WP16-10 Raj Kettimuthu, Computing and Networking Challenges in Supporting Streaming Applications

WP16-11 Shweta Prabhat Khare, Distributed Reactive Stream Processing

WP16-12 Scott Klasky, Stream Processing for Remote Collaborative Data Analysis

WP16-13 Kerstin Kleese Van Dam, Reliable Performance for Streaming Analysis Workflows

WP16-14 Andre Luckow, Pilot-Streaming: Design Considerations for a Stream Processing Framework for High-Performance Computing

WP16-15 Andre Martin, Elastic and Secure Energy Forecasting in Cloud Environments

WP16-16 Benji Maruyama, Autonomous Experimentation Applied to Carbon Nanotube Synthesis

WP16-17 Nina Mishra, Robust Random Cut Forest Based Anomaly Detection on Streams

WP16-18 Klaus Mueller, Mining Behavior Patterns in Streaming Multivariate Data

WP16-19 Srinivasan Parthasarathy, Ego-net Sketching for Streaming Graph Analytics

WP16-20 Brian Quiter, Radiological Search – a Long-Standing Streaming Application

WP16-21 Karthik Ramasamy, Streaming in Practice

WP16-22 Carlos Varela & Stacy Patterson, Steering Complex Systems Using a Dynamic, Data-Driven Modeling Approach

WP16-23 Matt Wolf, Rethinking Streaming System Construction for Next-Generation Collaborative Science

WP16-24 John Wu, Connecting Large Experimental Facility and Computing Facility with Streaming Analytics

WP16-25 Shinjae Yoo, Streaming Manifold Learning and DOE Applications

WP16-26 Dantong Yu, Deep learning for analyzing NSLS-II Data Stream

A.3 STREAM2015 References

STREAM2015 Workshop Presentations

P15-1. Roger Barga, *Kinesis: Data Stream Processing Services* [pdf]

P15-2. Pete Beckman, *Array of Things* [pdf]

P15-3. Jelena Pjesivac-Grbovic, *Google Cloud Dataflow* [pdf]

P15-4. Justin Wozniak, *Streaming, Storing, and Sharing Big Data for Light Source Science* [pdf]

P15-5. Alex Szalay, *Streaming Problems in Astrophysics* [pdf]

P15-6. Bojan Nikolic, *Data Processing for the Square Kilometre Array Telescope* [pdf]

P15-7. Vladimir Braverman, *Streaming Algorithms for Cosmological Simulations and Beyond* [pdf]

P15-8. Robert Brunner, *Breaking Data Science* [pdf]

P15-9. Yanif Ahmad, *Accelerating Scientific Discovery through Data-Driven Control and Real-Time Analytics* [pdf]

P15-10. Erik Paulson and Samuel Hamilton, *IIOT at Johnson Controls* [pdf]

P15-11. John Wu, *Baseline in Streaming Data Analysis: What, Why, and How?* [pdf]

P15-12. Madhav Marathe, *Real-Time Network Science* [pdf]

P15-13. Judy Qiu, *Parallel Clustering of High-Dimensional Social Media Data Streams* [pdf]

P15-14. Von Welch, *Urban Sensor Data Privacy Issues: Findings of the Array of Things (AoT)* [pdf]

P15-15. Kerstin Kleese Van Dam, Analysis and Decision Making in Big Data Environments Analysis in Motion (AIM) [pdf]

P15-16. Shrideep Pallickara, *Real-Time Stream Processing For Sensing Environments* [pdf]

P15-17. Martin Swany, *In-Network Processing for Streaming Data* [pdf]

P15-18. Dan Fay, *Streaming on Microsoft Azure* [pdf]

P15-19. Supun Kamburugamuve, *Streaming Applications for Robots with Real Time QoS* [pdf]

P15-20. Andre Martin, *StreamMapReduce: When Stream Processing crosses MapReduce* [pdf]

P15-21. Ilkay Altinas, *Dynamic Data-Driven Applications using Workflows as a Programming Model for Scalable and Reproducible Streaming and Steering* [pdf]

P15-22. Milinda Pathirage, *Distributed Streaming SQL* [pdf]

P15-23. Stefano Markidis, *Streaming Programming Systems for Exascale* [pdf]

P15-24. Joshua Suetterlein, *Toward a Unified HPC and Big Data Runtime* [pdf]

P15-25. Carlos Varela, *Dynamic Data-Driven Avionics Systems: Inferring Failure Modes from Data Streams* [pdf]

P15-26. Torre Wenaus, *Science-Aware Dynamic Data Delivery at the Exascale* [pdf]

P15-27. Maryam Rahnemoonfar, *Real time Oil and Gas source Identification using Unmanned Aerial Systems* [pdf]

P15-28. Xiangbo Li, *CVS: A Cost-Efficient and QoS-Aware Cloud Video Streaming* [pdf]

P15-29. David Crandall, *Streaming Deep Learning for Robotics and Mobile Cameras* [pdf]

STREAM2015 Workshop White Papers

WP15-1. Streaming Algorithms for Astronomic Simulations; Authors: Vladimir Braverman, Alex Szalay; Institution: Johns Hopkins University

WP15-2. StreamMapReduce: When Stream Processing crosses MapReduce; Authors: André Martin, Andrey Brito, Christof Fetzer; Institution: Technische Universität Dresden, Dresden, Germany; Universidade Federal de Campina Grande, Campina Grande, Brazil

WP15-3. Dynamic Data-Driven Applications using Workflows as a Programming Model for Scalable and Reproducible Streaming and Steering; Authors: Ilkay Altintas; Institution: UC San Diego

WP15-4. Toward a Unified HPC and Big Data Runtime; Authors: Joshua Suetterlein, Joshua Landwehr, Joseph Manzano, Andres Marquez; Institution: University of Delaware, Pacific Northwest National Lab

WP15-5. Streaming, Storing, and Sharing Big Data for Light Source Science; Authors: Justin M. Wozniak, Kyle Chard, Ben Blaiszik, Michael Wilde, Ian Foster; Institution: Argonne National Laboratory, University of Chicago

WP15-6. Accelerating the Experimental Feedback Loop: Data Streams at the APS; Authors: Ian Foster, Tekin Bicer, Raj Kettimuthu, Michael Wilde, Justin Wozniak, Francesco de Carlo, Ben Blaiszik, Kyle Chard, Francesco de Carlo, and Ray Osborn; Institution: Argonne National Laboratory

WP15-7. Accelerating Scientific Discovery through Data-Driven Control and Scalable Real-Time Analytics; Authors: Ben Ring, Yanif Ahmad, Tom Woolf; Institution: Johns Hopkins University

WP15-8. Real time Oil and Gas source Identification using Unmanned Aerial Systems; Authors: Maryam Rahnemoonfar; Institution: Texas A & M University - Corpus Christi

WP15-9. Analysis in Motion (AIM); Authors: Kerstin Kleese van Dam, Mark Greaves, Rob Jasper, Nigel Browning; Institution: Pacific Northwest National Laboratory, Brookhaven National Laboratory

WP15-10. Baseline in Streaming Data Analysis: What, Why, How?; Authors: John Wu; Institution: Lawrence Berkeley National Laboratory

WP15-11. Streaming Programming Systems at Exascale; Authors: Stefano Markidis, Ivy Bo Peng, and Erwin Laure; Institution: KTH Royal Institute of Technology

WP15-12. Science-Aware Dynamic Data Delivery at the Exascale; Authors: M. Ernst, A. Klimentov, N. Malitsky, T. Wenaus, P. Calafiura, D. Malon, R. Mount, S. Jha, K. De; Institution: Brookhaven National Laboratory, Lawrence Berkeley National Laboratory, Rutgers University, Argonne National Laboratory, University of Texas at Arlington, Stanford Linear Accelerator Center

WP15-13. Cloud-Based Video Streaming for Energy and Compute-Limited Thin Clients; Authors: Xiangbo Li, Mohsen Amini Salehi, Magdy Bayoumi; Institution: The Center of Advanced Computer Studies (CACS), University of Louisiana Lafayette

WP15-14. Urban Sensor Data Privacy Issues: Findings of the Array of Things (AoT) Privacy Br eakout Group; Authors: Von Welch, Charlie Catlett; Institution: Indiana University, Argonne National Laboratory

WP15-15. Dynamic Data Driven Avionics Systems; Authors: Carlos Varela; Institution: Rensselaer Polytechnic Institute

WP15-16. Fast Data Management with Distributed Streaming SQL; Authors: Milinda Parthirage and Beth Plale; Institution: School of Informatics and Computing, Indiana University

WP15-17. Streaming Application for IOT with Real Time QoS; Authors: Supun Kamburugamuve & Geoffrey Fox; Institution: School of Informatics and Computing, Indiana University

Appendix B: STREAM2016 Agenda

STREAM 2016 Agenda
March 22, 2016

8:45 AM - 9:30 AM
Opening Session

- Introductions around the room
- Stream Recap 2015 (Geoffrey) [15 mins]
- Stream 2016 Goals [10 mins]
- Open Discussion

9:30 AM -10:30 AM
Application [15 mins per talk + 15 mins discussion]
Moderator: Geoffrey Fox

- Craig Tull (TBD)
- Vakhtang Tsulaia (TBD: Streaming in ATLAS)
- Nathan Tallent (Processing large scale streaming data from high energy physics workflows)
- Open Discussion

10:30 AM - 11:00 AM: Break

11:00 AM - 11:45 AM
Applications/Middleware [10 mins per talk + 15 mins discussion]
Moderator: Kerstin Kleese Van Dam

- Marty Humphrey (Leveraging Public Clouds for DOE Environmental Streaming Data)
- Brian Quitter (Radiological Search – A Long-standing Streaming Application)
- Benji Maruyama and Rahul Rao (Autonomous Experimentation Applied to Carbon Nanotube Synthesis)
- Open Discussion

11:45 AM - 12:30PM: Lunch

12:30 PM - 1:10 PM: Remote Talks (Salman Habib and Dean Williams)/Discussions

1:10 PM - 2:15 PM: Breakout, Application Needs, Challenges, Gaps
Breakout Leads: Matthew Wolfe, Craig Tull, Katherine Riley, Shrideep Pallickara

2:15 PM - 2:45 PM: Summary of Breakouts

2:45 PM - 3:15 PM: Break

3:15 PM - 4:00 PM

Streaming Systems for Collaborative Science: [10 mins per talk + 15 discussion]
Moderator: Dennis Gannon

- Matt Wolf (Rethinking streaming system construction for next-generation collaborative science)
- Kerstin Kleese Van Dam (Reliable Performance for Streaming Analysis Workflows)
- Scott Klasky (Stream Processing for Remote Collaborative Data Analysis)
- Open Discussion

4:00 PM - 5:15 PM
Streaming in Industry [15 mins per talk + 30 mins Panel/Discussion = 75 mins]
Moderator: Amy Apon

- Karthik Ramasamy (Streaming in Practice)
- Eugene Kirpichov, Google, TBD
- Roger Barga (Moving Towards Streaming-Data Analysis)
- Panel/Discussion

5:15 PM - 5:30 PM: Wrap-up

March 23, 2016

8:30 AM - 9:30 AM
Network/Facility [10 mins per talk + 20 discussion]
Moderator: Shantenu Jha

- Dantong Yu (Deep learning for analyzing NSLS-II data stream)
- John Wu (Connecting large experimental facility and computing facility with streaming analytics)
- Raj Kettimuthu (Computing and networking challenges in supporting streaming applications)
- Dimitrios Katramatos (Streaming Data Analysis on the Wire)
- Open Discussion

9:30 AM - 10:45 AM
Programming Model/DDDAS Session [10 mins per talk + 25 discussion]
Moderator: Manish Parashar

- Jack Dennis (Programming Model and Architecture for Real Time Streaming)
- Shweta Khare (Distributed Reactive Stream Processing)
- Nina Mishra (Robust Random Cut Forest Based Anomaly Detection On Streams)
- Carlos Varela (Steering Complex Systems using a Dynamic, Data-Driven Modeling Approach)
- Mohsen Amini (HLSAAS: HIGH-LEVEL LIVE VIDEO STREAMING AS A SERVICE)
- Open Discussion

10:45 AM - 11:00 AM Break

11:00 AM - 12:00 PM
Applied Maths-Statistics/Algorithms: [10 mins per talk + 20 discussion]
Moderator: Carlos Varela

- Chen Ding (Timescale Stream Statistics for Hierarchical Management)
- Klaus Mueller (Mining Behavior Patterns in Streaming Multivariate Data)
- Shinjae Yoo (Streaming Manifold Learning and DOE Applications)
- Srini Parthasarthy (Ego-net Sketching for Streaming Graph Analytics)
- Open Discussion

12:00 PM - 12:40 PM Lunch

12:40 - 1:00 Applications

Alex Szalay (Streaming in Astronomy)

1:00 PM - 2:00 PM
Middleware and Software Systems: [10 mins per talk + 20 discussion]
Moderator: Scott Klasky

- Gagan Agrawal (Can Commercial BigData Ideas Benefit Analysis of Instrument Data?)
- Andre Martin (Elastic and Secure Energy Forecasting in Cloud Environments)
- Andre Luckow (Pilot-Streaming: Design Considerations for a HPC Stream Processing Framework)
- Geoffrey Fox (WebPlotViz: Browser Visualization of High Dimensional Streaming Data with HTML5)
- Open Discussion

2:00 PM - 2:45 PM: Breakout, Research Directions

- What is state of art?
- What are research directions needed by DOE applications, other applications and industry?

2:45- 3:00 Break

3:00 PM - 4:00 PM Summary of Breakout and Discussions

- Community Engagement, Report Writing

Appendix C: STREAM2016 Session Summaries

Appendix: Session Summaries

Day 1 March 23 2016
8:45 AM - 9:30 AM Opening Session
Geoffrey Fox: Summary of Streaming Data Workshop STREAM2015 October 27-28 2015

Slides: http://dsc.soic.indiana.edu/presentations/STREAM2015-Overview-Mar22-2016.pptx
This talk reviewed the earlier workshop held October 27-28 2015 in Indianapolis. This had 43 attendees, 17 Workshop white papers (from call for participation) and 29 Presentations (28 with slides; 23 with videos). The workshop website http://streamingsystems.org/ has background material plus STREAM2015 resources and the final Report of STREAM2015 workshop at http://streamingsystems.org/stream2015finalreport.html. This workshop covered the field broadly and there was a consensus that it usefully brought together an unusual interesting set of participants and there was enthusiasm for continuing such activities. The workshop covered a different set of topics to STREAM2016 with technology, applications and education. Industry was covered with Amazon, Google and Microsoft from technology side and Johnson Controls from the "Industrial Internet of Things IIoT". A classification of algorithms into 8 categories with different characteristics such as event size, synchronicity, time & length scales, was given

1. Industrial Internet of Things, Cyberphysical Systems, DDDAS, Control
2. Internet of People: wearables
3. Social media, Twitter, cell phones, blogs, e-commerce and financial transactions
4. Satellite and airborne monitors, National Security: Justice, Military
5. Astronomy, Light and Neutron Sources, TEM, Instruments like LHC, Sequencers
6. Data Assimilation
7. Analysis of Simulation Results
8. Steering and Control

The final report reviewed current technology solutions with a plethora of "local point" solutions but few end-to-end general streaming infrastructures outside open sourced big data systems such as Apache Spark, Flink, Storm, Heron and Samza, whilst in their application class, commercial solutions such as Kinesis and MillWheel were most advanced. Issues identified included that current XSEDE and DoE infrastructure not optimized for streaming data and the importance of issues in distributed computing, such as performance, fault-tolerance, and dynamic resource management. The interface of HPC and streaming was extensively discussed. Novel algorithm issues including online and sampling methods and the reduction of $O(N^2)$ algorithms to O(NlogN) were highlighted. The importance of benchmarks, application collections and streaming software system and algorithm libraries were stressed as was the need for infrastructure optimized for streaming applications. Near term collection and prototyping activities in these areas were identified.

Shantenu Jha: Goals of STREAM2016 Meeting

This talk provided some background about participation numbers, technical organization of the workshop and design of sessions The talk also highlighted the main focus and objectives of the workshop and compared them to STREAM2015.

https://docs.google.com/presentation/d/12CWX6vOae2DGRvGJc157YBz-LHhRU2F76txkf0yZNEM/edit

9:30 AM -10:30 AM Applications

Craig Tull: Real-time Streaming Analysis for BES User Facilities

Slides: http://streamingsystems.org/Slides/cet20160322-stream2016.pptx.pdf

Tull described use of NERSC for analysis of interesting different cases: ALS Advanced Light Source at LBNL with remote experiment control enabled, material science with grazing-incidence small-angle X-ray scattering, GISAXS and Daya Bay neutrino experiment in China. The technology highlighted was SPADE developed in IceCube, and used in Daya Bay and ALS data movement while XSWAP managed complex DAG-based workflows with around 50 nodes. Publish-subscribe technology RabbitMQ improved schedul- ing. Tull divided streaming problems by the processing delay tolerated:

- **Overnight** (eg. telescopes, day shift experiments): Plan campaign for next shift/day
- **Hourly** (eg. stable, long-term HEP experiments): Detect problems; Maintain steady-state data taking
- **Minutes** (eg. time-resolved, in-situ experiments): Follow experiment evolution; Verify data quality
- **Instantaneous** like a "software" microscope

Vakhtang Tsulaia: Streaming in ATLAS

Slides: http://streamingsystems.org/Slides/ATLASStreaming.pdf (STREAM2016)
Slides: http://streamingsystems.org/Presentations/Torre%20Wenaus.pdf (STREAM2015)
Paper: http://streamingsystems.org/WP/ScienceAware.pdf (STREAM2015)
Tsulaia described the ATLAS LHC experiment with 3000 scientists and 1200 students from 38 countries. It gathers 1 PB/sec of data filtered to 1-2GB/sec recorded. The experimental computing environment is instantiated on 140 heterogeneous worldwide resources enabled by excellent networking. ATLAS data processing is moving from classic file based grids to an Event Service (ES), which is operational today, with quasi-continuous event streaming through worker nodes and exploiting opportunistic resources and minimizing local storage demands. They use the PanDA Distributed Workload Manager optimized (Yoda) for use on HPC systems. It features whole-node scheduling, remote I/O, and use of object stores. A next step is to enhance the event service ES to ESS or Event Streaming Service.

Nathan Tallent: Processing Streaming Data In High Energy Physics Workflows

Slides: http://streamingsystems.org/Slides/stream-2016-slides.pdf

Tallent described Belle 2 at KEK in Japan, which gathers 25 PB/year raw data. They have developed hierarchical scheduling to mitigate contention, and reduce power consumption. This uses a sophisticated, analytical model to predict execution time fed by a provenance engine to gather performance metrics. Further data transfer is optimized by prefetching.

11:00 AM - 11:45 AM Applications/Middleware

Marty Humphrey: Leveraging Public Clouds for DOE Environmental Streaming Data

Paper: http://streamingsystems.org/Papers/Humphrey.pdf
Slides: http://streamingsystems.org/Slides/Humphrey%20Stream%202016.pdf
Combined summary given below

Brian Quiter: Radiological Search – A Long-standing Streaming Application

Paper: http://streamingsystems.org/Papers/Quiter.pdf
Slides: http://streamingsystems.org/Slides/Quiter_STREAM2016-noVideo.pdf
Combined summary given below

Benji Maruyama and Rahul Rao: Autonomous Experimentation Applied to Carbon Nanotube Synthesis
Paper: http://streamingsystems.org/Papers/Maruyama.pdf
No slides available. Combined summary given below

a) 3 Use Cases

The use cases presented in this session explored streaming data analysis in multi-source data environments, with varying levels of geographic distribution and heterogeneity of data types and formats. They furthermore explored to varying degrees the possibility to utilize the results of the streaming analysis for autonomous decision making and steering (of the data collection) by the system.

AmeriFlux and FLUXNET (Humphrey) represent an Internet of Things scenario, where there are many sensors out in the field that create atmospheric measurement records of varying quality and heterogeneous formats. There is a desire to move from sporadic data submissions from the site PIs responsible for one or more sensors, to an automated and streaming data collection, cleaning and analysis environment that would shorten the time from data collection to data product delivery to the scientific researcher. The Radiological search (Quiter) is similar in nature. Sensors deployed on aircrafts (ARES) and trucks (RadMAP) collect ambient gamma-rays and neutron measurements to detect, localize and identify possible threats against a background of benign radioactivity. Investigations are now underway to see if the streaming analysis of this data could be improved, if information from other contextual sensors (incl. high-definition video, Lidar, hyperspectral imagery etc.) could be integrated in real time, leading to a streaming multi-source, heterogeneous data type scenario. Furthermore there is a desire to test if computationally steered measurements based on the streaming analysis would lead to more efficient data gathering and more effective detection. The Air Force Research Laboratory (Maruyama) presented a variation on this theme. Multi-sensor measurements from a single instrument are used in an autonomous research system to design, execute and analyze experiments. The system is currently applied to understand and control the synthesis of single wall carbon nanotubes, to optimize their growth rate.

b) Current Methods - what is being done now, what is available

All projects are only starting out on their path to streaming data analysis, most data analysis today is still done post hoc on static data. The more established projects such as the advanced research system at the Air Force Research Laboratory and initial work at RadMAP are based on complete custom solutions, whereas later efforts such as those around AmeriFlux aim to leverage commercial solutions such as Amazon Kinesis and Lambda for their research infrastructure.

ARES and RadMap

For ARES streaming algorithms have been developed to highlight spectral changes or matches to threat signatures. The greatest challenge in developing these algorithms is the low statistics available and the naturally changing radiological environment. Data rates: Radiation sensors 500GB/hr after online processing and Object tracking 7xHD video camera – 2TB/hr. RadMAP data is currently only processed after collection.

c) Future Challenges/R&D needs

Multi-source data analysis infrastructure – Needed ability to handle geographically distributed data streams effectively and in sync through the analysis pipeline. When are public cloud infrastructures the right solution, when are custom build infrastructures needed?

Multi-source, multi-data type analysis algorithms and frameworks – multi-source heterogeneous data analysis is a research challenge in its own right, today most of these analytics projects (even post hoc) are one off hero efforts, due to the need to overcome semantic, syntactic, representational and scale differences between the different data streams, as well as the need to provide mappings between the varying information contents based on the specific scientific scenario.

Validation and Verification – There is a need to provide test environments in which analysis algorithms can be evaluated against existing solutions in terms of speed, accuracy and added value. Furthermore such a test range would be useful to demonstrate and quantify the impact of different data streams on the accuracy and value of the analysis result for the subsequent decision making process. In addition however there is a need to validate the analytical and decision model's at runtime, are they still correct or do they need adjustment/ retraining.

Quantifying and Reducing Uncertainty – as streaming analysis works on incomplete data and usually only has one pass through the data, there are varying levels of uncertainty associated with the analysis results throughout the process runtime. To date there are no methods available to sufficiently quantify this uncertainty or strategies proposed to reduce it were required e.g. through introduction of additional data sources.

Human-Machine partnering – Effective interaction of human and computers is key in all the systems, but in particular in those with increased autonomy, here the researcher needs to be able to effectively express research goals, the system needs to be able to communicate results and request for additional information effectively and both have to be able to iterate on future strategies.

12: 30 PM - 1:10 PM: Remote Talks

Salman Habib: Streaming Data in Cosmology

Slides: http://streamingsystems.org/Slides/streaming_data_cosmology_habib_stream_2016.pdf

This remote presentation described and contrasted the challenges of analyzing cosmologically oriented simulation and observation data examined for "interesting" events. A mix of online (in situ for simulations) and offline analysis is needed with complex workflows. The observational data comes from Cosmic Microwave Background, Optical and Radio sources. Projects include PTF (Palomar Transient Facility), DES (Dark Energy Survey) and the future LSST (Large Synoptic Survey Telescope) with current data rates at about 500 GB/night while the future LSST (2022 operation) can go up to 20TB/night, with about 10K alerts/night. Machine learning to identify events is an active research area. Today Cosmological simulation offline data flows already require ~PB/week capability, The talk concluded with the Extreme-Scale Analytics Systems (EASy) Project . This combines aspects of High Performance Computing, Data-Intensive Computing, and High Throughput Computing with an initial focus on cosmological simulations and surveys. Key elements of this project are:

- Software Stack: Run complex software stacks on demand (containers and virtual machines). Are clouds useful?
- Resilience: Handle job stream failures and restarts
- Resource Flexibility: Run complex workflows with dynamic resource requirements
- Wide-Area Data Awareness: Seamlessly move computing to data and vice versa; access to remote databases and data consistency

- Automated Workloads: Run automated production workflows
- End-to-End Simulation-Based Analyses: Run analysis workflows on
- simulations and data using a combination of in situ and offline/coscheduling approaches

Dean Williams: The Earth System Grid Federation (ESGF)
Slides: http://streamingsystems.org/Slides/ESGF_2016_STREAM_Presentation.pdf

The Earth System Grid Federation (ESGF) is a multi-agency collaboration of around 40 organizations to facilitate and empower the study of climate using data management, stewardship and curation. One important collection is CMIP5 -- the Coupled Model Intercomparison Project ESGF maintains a software infrastructure for management, dissemination, and analysis of simulation and observational climate data. There is a streaming project for looking at multi-resolution climate simulation ensembles. ESGF sets networking best practices into place to effectively transport tens of petabytes of climate data supporting 1PB/month (rising to 4PB) of sustained disk-to-disk data transfer between ESGF primary data centers.

3:15 PM - 4:00 PM Streaming Systems for Collaborative Science:
Matt Wolf: Rethinking streaming system construction for next-generation collaborative science
Paper: http://streamingsystems.org/Papers/Wolf.pdf
Slides: http://streamingsystems.org/Slides/STREAM_2016_MWolf.pdf

Matthew Wolf described the work Georgia Tech has done with Oak Ridge and other collaborators on middleware support for scientific data stream analysis.

The use case described here is from combustion science. The experimental framework is a combustion scenario designed to understand the dynamics of fuel mixes, speeds and acoustic interactions. Particles are injected into the combustion and sets of cameras capture images at time intervals allowing velocity fields to be calculated. The streaming data is the images which are processed first to determine if the data is valid. If not, the experiment must be rerun. If it is valid is it interesting? Does it address the part of the physical parameter space that is of interest? If not, then the experiment must be refined and the process is repeated. There is another level of streaming data here. This includes cross-stream computations that relates the experiment to previous runs and simulations as well as collaboration between multiple sites with relevant data that must be included in the analysis.

The current methods are traditionally based on ad hoc "bespoke" software that was designed to fit the very specific needs of the experiments. Wolf makes the case that a general stream analysis system is not going to exist that will cover this case as well as others. A better solution is to deploy a reusable toolkit of components that can be assembled to build application-level overlay networks with embedded computation. The toolkit they use is EVPath http://evpath.net. Key concepts from EVPath are embedded in ADIOS as FlexPath, a high performance I/O library from Oak Ridge. ADIOS facilitates some basic workflow capabilities for managing the processing of data from HPC systems experiments and FlexPath provides a basic publish/subscribe capability to the ADIOS workflow model.

The research challenge they identify is that interactivity involves more than human-in-the-loop. Advanced middleware must enable delegation of control. Change management is also becoming more critical as the technology is evolving very rapidly.

Kerstin Kleese Van Dam (Reliable Performance for Streaming Analysis Workflows)
Paper: http://streamingsystems.org/Papers/Kleese%20van%20Dam.pdf
Slides: http://streamingsystems.org/Slides/Kleese%20-%20STREAM%202016%20-%20IPPD.pdf

Kerstin Kleese Van Dam makes the important observation that that the workflow systems managing the stream analytics of time-critical experiments can be complex and success of the experiment depends upon reliable performance of the overall system.

The use case is "In Operando catalysis experiments". More specifically, this involves the steering of high end electron microscopy experiments where a beam of electrons is transmitted through an ultra-thin specimen, interacting with the specimen as it passes through. These experiments can generate atomic resolution diffraction patterns, images and spectra under wide ranging environmental conditions. In-situ observations with these instruments, were physical, chemical or biological processes and phenomena are observed as they evolve. These experiments generate from 10GB-10's of TB (e.g. at BNL) of data per at rates ranging from 100 images/sec for basic instruments to 1600 images/sec for state of the art systems. To optimize the scientific outcome of such experiments it is essential to analyze and interpret the results as they are emerging. It is essential that the workflow system reliably deliver optimal performance, especially in situations where time-critical decisions must be made or computing resources are limited.

The current systems in use include the Analysis in Motion framework developed by PNNL, but the challenge that is presented here is to enact the workflow in a way that yields reliable performance when the execution of the workflows, frequently composite applications built from loosely coupled parts, running on a loosely connected set of distributed and heterogeneous computational resources. Each workflow task may be designed for a different programming model and implemented in a different language, and most communicate via files sent over general purpose networks.

The DOE ASCR funded Integrated End-to-End Performance Prediction and Diagnosis for Extreme Scientific Workflows (IPPD) project address this performance reliability project on a number of fronts. They are exploring a combination of empirical studies and performance modeling. In addition they are looking at the challenge of optimizing the file I/O and data transfers. The result will be a toolkit that will optimize workflow performance through improved scheduling that will reduce contention on shared, distributed resources.

Scott Klasky: Stream Processing for Remote Collaborative Data Analysis
Paper: http://streamingsystems.org/Papers/Klasky.pdf
Slides: http://streamingsystems.org/Slides/Stream-2016-klasky.pdf

The use case is the set of international fusion energy projects such as the International Thermonuclear Experimental Reactor (ITER), Korea Superconducting Tokamak Advanced Research (KSTAR), and National Spherical Torus Experiment (NSTX). In these projects a team of scientists have to be present at the facilities to monitor the progress of the on-going data collection, adjust the control settings, and prevent catastrophic events, while most others access the data remotely. It is truly a collaborative data analysis challenge. The most difficult part is handling the growing data sizes over the network. What is lacking is a software system that allows scientists to quickly and conveniently compose complete analysis tasks, manage the necessary data movement, execute the specified tasks, and provide timely feedback to the users.

The technology used is a new system called ICEE which is composed of the ADIOS I/O and EVPath. The key contribution of ICEE is the addition of memory-to-memory data streaming over a wide variety of underlying protocols. ICEE presents a uniform abstraction for build the data analysis workflow using another tool called DataSpaces.

The greatest research challenge is defining the abstractions that characterize a reasonable subset of use cases that can be used to create a toolkit that can be used to build efficient, high performance solutions. The papers in this section describe several such efforts. It is clear that the scientific use cases will depend more heavily on workflow

management capabilities and data management that is both file and stream based. Another challenge is the ability for collaboration. This takes two forms. One case is allowing multiple teams of researchers to tap into the "stream" to do different experimental analyses. The other case is when one team of specialists depends upon the work of another "upstream" team who have been responsible for various preprocessing steps. In both cases the participants may be very geographically distributed.

4:00 PM - 5:15 PM Streaming in Industry
Karthik Ramasamy: Streaming in Practice

Paper: http://streamingsystems.org/Papers/Ramasamy.pdf

Slides: http://streamingsystems.org/Slides/heron-stream-2016-workshop.pdf

The presentation by Dr. Karthik Ramasamy, focused on the newest middleware from Twitter, called Heron. Twitter is synonymous with real time, and Twitter generates tens of billions of events per hour with over 315 million monthly active users. With the requirement of processing this very large scale real time data, Twitter saw the need to develop an entire new distributed stream processing engine. Heron is the next generation of streaming systems at Twitter. Heron was initiated by Ramasamy, who also oversaw its development. The language of development, C++, is more likely to promote a strong developer community as compared to the language used for its predecessor (Storm), which was Clojure. Heron is designed to provide: ease of development and troubleshooting, efficiency and performance, scalability and reliability, compatibility with Storm, a simplified and responsive user interface, and capacity allocation and management. The talk described the data model, which is based on a directed acyclic graph for representing real time computing. The talk also described mechanisms in the middleware, such as back pressure and load shedding, which reduce data loss while keeping the data rate high. The talk included results from experimental testing. Heron has been in production use in Twitter, has been highly optimized, and will become open source. A lively discussion about the type of applications and data most suited for Heron ensued.

Eugene Kirpichov: Dataflow / Apache Beam - A Unified Model for Batch and Streaming Data Processing

Slides: http://streamingsystems.org/Slides/Eugene%20Kirpichov%20-%20STREAM%202016%20Dataflow%20and%20Apache%20Beam.pdf

The Apache Beam presentation by Eugene Kirpichov, centered around four critical questions that all data processing practitioners must attempt to answer when building data pipelines: What results are calculated? Where in event time are results calculated? When in processing time are results materialized? How do refinement of results relate? The talk provided the historical and technical foundation for the current unified model for batch and stream data processing, and discussed some earlier data processing tools and methods, including MapReduce, Spark, and Google Cloud Dataflow. Spark provides a unified batch and streaming engine and supports answer for the four questions. However, Spark lacks a formal notion of event-time windowing, which forces the intermingling

Day 2 March 23 2016
8:30 AM - 9:30 AM Network/Facility
Dantong Yu (Deep learning for analyzing NSLS-II data stream)

Paper: http://streamingsystems.org/Papers/Yu.pdf

Slides: http://streamingsystems.org/Slides/DantongYu_Deep_learn_Data_Streaming.pdf

The National Synchrotron Light Source (NSLS-II) provides extremely bright x-rays for basic and applied research in biology and medicine, materials and chemical sciences, geosciences and environmental sciences, and nanoscience; it is up to 10,000 as powerful as its predecessor. It produces large volumes of complex data. The objective is be able to provide deep learning approaches to provide a first-line of analysis on the diffraction images made available. Further, there is a need to provide automated materials discovery across many synchrotron beamlines (referred to as Multimodal Analysis). There are multiple levels at which deep-learning approaches can play a role: at the finest

level there is the need to determine/detect features; further up, there is a need to extract physical process/phenomenon from a sequence of images. At the highest-level, there is a need to determine physically meaningful trends. The infrastructure and algorithms must be such that the velocity of processing must be commensurate with that of data generation, so that machine-learning can become a critical component of automated materials discovery.

John Wu: Connecting large experimental facility and computing facility with streaming analytics
Paper: http://streamingsystems.org/Papers/Wu.pdf
Slides: http://streamingsystems.org/Slides/Wu-TechnologySamples.pdf

This talk focussed on the technology need to support a range of distributed streaming analytics. Use cases ranging from near-real time feature detection, to segmenting microscopy images so as to identify cancerous cells in tissue images were presented. Requirements along the 5Vs were provided: (i) velocity: reduce data access latency, (ii) volume: reduce data volumes transferred by moving analysis smartly, (iii) variety: enable multiple streams of data concurrently, (iv) veracity: trade-offs between accuracy and performance, and (v) value: support interactive analysis. Some partial and existing technology solutions to these requirements were discussed – ranging from novel data reduction approaches based on statistical similarity to using indexing to locate distributed data efficiently. The talk presented some open questions and issues pertaining to algorithms, systems and networking needs to support the use cases discussed.

Raj Kettimuthu: Computing and networking challenges in supporting streaming applications
Paper: http://streamingsystems.org/Papers/Kettimuthu.pdf
Slides: http://streamingsystems.org/Slides/160323-Stream2016-Kettimuthu.pdf

Data streaming applications require compute resources at a specific time, for a specific period with a high degree of reliability. Such requirements are hard to meet on current HPC systems, which are typically batch-scheduled under policies in which an arriving job is run immediately only if enough resources are available, and is queued otherwise. Although cloud systems address some of these challenges, the use of leadership class computing facilities remains an important requirement. The talk proposed a series of considerations associated with the question: What changes will be required to the scheduling policies, architecture, and implementation of next-generation (and current) supercomputers if they are to support streaming science workloads effectively? What kind of allocation and charging policies are suitable to accommodate real-time jobs? From a network perspective, in order to overcome the limitations of the batch-queue systems, a congestion-free network path is required to stream data from data source to compute resource at a rate that is same as the data generation rate.

Dimitrios Katramatos: Streaming Data Analysis on the Wire
Paper: http://streamingsystems.org/Papers/Katramatos.pdf
Slides: http://streamingsystems.org/Slides/Katramatos-STREAM2016.pdf

As data volumes and distribution increase, the proportion of data that is in transit increases. Thus, early processing could in principle provides real- time/near real-time information that can be used to speed up the decision processes, which motivates the question: Is it feasible to devise a framework for data analytics on the wire, i.e., utilizing capabilities of the network infrastructure? Whereas this question has been asked in the past, both the scale of data and the sophistication of networks provide a new urgency and case to revisit the possibilities. For example, nowadays, network infrastructure includes mechanisms that can be programmed to recognize specific data flows based on given criteria. Can this be utilized to intercept and seamless redirect flows to processing sub-systems where data is subjected to desired processing.

9:30 AM - 10:45 AM Programming Model/DDDAS Session

Jack Dennis: Programming Model and Architecture for Real Time Streaming

Paper: http://streamingsystems.org/Papers/Dennis.pdf

Slides: http://streamingsystems.org/Slides/Fresh%20Breeze%20Streams.pdf

This presentation/whitepaper described the Fresh Breeze project, which is developing a programming model and system architecture for real time streaming applications. Fresh Breeze supports producer/consumer parallelism for streaming computations and data parallel processing for classical HPC applications. User programs are written in funJava, a functional variant of Java; the Fresh Breeze com- piler converts funJava programs into codelets for execution by a simulated Fresh Breeze multi-core pro- cessor. The Fresh Breeze multi-core architecture uses an instruction set that directly supports operations on trees of chunks and the scheduling of tasks for codelet execution. The presentation noted that there is no operating system or runtime software to add to overhead and increase energy consumption. A simula- tion model of a multi-core Fresh Breeze processor has demonstrated linear speedup for up to at least 256 processing cores for matrix multiplication. The team is exploring applications of funJava and the Fresh Breeze system architecture. The presentation was followed by a lively discussion about the architecture (particularly the memory structure) and its applicability to various data streaming scenarios. Distributed

Shweta Khare: Distributed Reactive Stream Processing

Paper: http://streamingsystems.org/Papers/Khare.pdf

This presentation/whitepaper explored open challenges in extending the reactive programming paradigm to distributed stream processing. This talk/whitepaper explored the challenges of the Internet of Things (IoT) paradigm and the resulting data streams, and the role of distributed Streams Processing Systems (DSPS) and Data Distribution Services (DDS). It identified resilience, responsiveness and elasticity as key attributes of a DSPS system. It also described a DSPS system that integrates a DDS with a reactive programming library and discussed the advantages of reactive programming for stream processing. Finally, the talk/whitepaper presented a research roadmap for reactive programming based DSPS as well as the state of the art in this space. The talk was followed by a discussion about many of the concept explored in the talk as well as about current practices.

Nina Mishra: Robust Random Cut Forest Based Anomaly Detection On Streams

Slides and Paper: not made public

This presentation/whitepaper explored anomaly detection in fast-moving data streams. The presentation/whitepaper noted that while anomaly detection has be explore extensively in the past, applying it to the vast quantities of data streaming from sensors, devices and the internet of things requires a different way of thinking about anomalies, which was the focus of the talk. The presented approach proposed the Robust Random Cut Tree data structure as an effective synopsis of a dataset, and developed a methodology for efficiently maintaining this data structure on a stream – a point is considered an anomaly if it has a large effect on the size of the robust random cut tree structure. The presentation/whitepaper presented an empirical study of the presented approach using two publicly available datasets. The first was the Washington DC daily bike rental dataset integrated with weather information such as windspeed/humidity/precipitation, as well as national holidays. In this study, anomalies corresponded to days with good weather where there are peaks in rentals, as well as windy and rainy days where there are lulls in rentals. The second used taxi ridership data from the NYC Taxi Commission, and considered a stream of the total number of passengers aggregated over a 30-minute time window. Anomalies in this data included holidays and snowstorms where ridership typically drops, as well as the NYC marathon and New Year's Eve when ridership typically peaks. The results suggested that the presented method not only has higher positive precision and positive recall, but also can detect anomalous time periods more quickly than isolation forest. The following discussion explored the

scalability of the approach to larger data sets.

Carlos Varela: Steering Complex Systems using a Dynamic, Data-Driven Modeling Approach

Paper: http://streamingsystems.org/Papers/Varela.pdf

Slides: http://streamingsystems.org/Slides/STREAM-PILOTS-DISTILL-March-23-2016-public.pdf

The research presented in this presentation/whitepaper was motived by dynamic data-driven applications and systems (DDDAS) that use models to control systems using a feedback loop, and where the data acquisition process itself may be controlled by steering the system resources allowing the modification of the models dynamically to incorporate recent observations. Motivating applications scenarios include online failure modeling and control in aircrafts, fire hazard monitoring and management, Internet of planes, etc. The presentation specifically explored a flight assistant system that incorporated dynamic data, for example to avoid bad weather while avoiding collisions and staying within capacity constraints. The talk discussed the Air France Flight 447 scenario and used it to motivate Dynamic Data-Driven Avionic, which uses a data-driven feedback loop to continuously analyze spatio-temporal data streams from airplane sensors, identify potential failure modes, and correct erroneous data, providing a new layer of logical redundancy in addition to existing physical redundancy for safer flight systems. A research program composed of a new mathematical formulation as well as the PILOTS programming language was presented. PILOTS enables declarative (high-level) definition of DDDAS data streaming application models (input-output relationships between data streams), error signatures, and error correction functions, and the PILOTS software detects specific (e.g., failure-induced) data errors based on signatures and corrects data before processing according to the application model. The evaluation presented using the AF447 data confirmed the effectiveness of the PILOTS approach and demonstrated how it could successfully detect and correct the airspeed sensor failure in this case after 5 seconds from beginning of the failure, with an overall error mode detection accuracy of 96.31

Mohsen Amini: HLSAAS: High-level Live Video Streaming as a Service

Paper: http://streamingsystems.org/Papers/Amini.pdf

Slides: http://streamingsystems.org/Slides/Mohsen-Amini-Streaming2016.pdf

This presentation/whitepaper focused on live video streaming services that are enabled by high-speed networks, and addresses the lack of high-level services and support for qualities demanded by the viewers and video providers. Specifically the talk addressed two challenges, the computational requirements of video processing services, and the real-time nature of video processing and the related QoS demands of live streaming viewers. To address these challenges and support a flexible range of high-level live video streaming services, the presentation/whitepaper described the High- level Live Streaming as a Service (HLSaaS) cloud-based architecture. HLSaaS can apply any high-level video-processing request on live video streams. Based on the request, it allocates computational resource from cloud to minimize the incurred cost while respecting the QoS demands of the viewers. The presentation/whitepaper also explored high-level live streaming services that can take advantage of the HLSaaS architecture, and described two such services, privacy aware live video streaming and live video transcoding. The presentation/whitepaper then presented an experimental evaluation of the scheduling of HLSaaS and demonstrated its effectiveness in reducing startup delay. The presentation concluded by outlining future research directions, including matching video types with heterogeneous cloud services, and integrating HLSaaS with content delivery networks and video on demand services. The presentation was followed by questions further exploring the details of the architecture as well as its broader applicability.

11:00 AM - 12:00 PM Applied Maths-Statistics/Algorithms

Chen Ding: Timescale Stream Statistics for Hierarchical Management

Paper: http://streamingsystems.org/Papers/Ding.pdf

Slides: http://streamingsystems.org/Slides/Chen-Ding-slides.pdf

This paper explores the consequence of computer system variations characterized by a time window. This could either be determined by user program needs or by external factors such as temperature. An example is given of in-memory key-value store Memcached hierarchical memory management optimization. A challenge is that the total number of windows is quadratic in the length of a stream. Linear time algorithms have been found and successfully used for online optimization.

Klaus Mueller: Mining Behavior Patterns in Streaming Multivariate Data

Paper:http://streamingsystems.org/Papers/Mueller.pdf

Slides: http://streamingsystems.org/Slides/STREAM2016.pdf

A behavior is defined as a time-varying pattern of a single variable seen in subsequences. Behaviors are used to determine dependencies, correlations, and possibly causations between multiple variables with possible use of subsequences or motifs. The visual analytics package *StreamVis* is applied to urban pollution data, so as to enable domain experts to mine, hypothesize, and validate multivariate behavior relationships. Issues raised include:

- What is appropriate distance or similarity metric that can be used to correlate data? Beyond Euclidean distance, can structural similarity, dynamic time warping, auto regression, or other distance metrics provide more powerful analytics??
- How to model concept drift -- —the evolution of data over time?
- How to deal with memory constraints that may enable only one-pass processing over massive data streams?
- How to use behaviors to summarize massive data, so that key multi-variate relationships can be visualized, discovered, stored, and evolved over time?
- What is the ideal (perhaps inherent) periodicity in the (potentially multi-scale) data? Can frequency and wavelet analyses be used to help data analytics?

Shinjae Yoo: Streaming Manifold Learning and DOE Applications

Paper: http://streamingsystems.org/Papers/Yoo.pdf

Three major use cases are introduced: materials science, climate science, and biology. In material sciences, Transmission Electron Microscopy (TEM) at the Center for Functional Nanomaterials (CFN) generates 3GB/s raw video streams (up to 1600 fps). Manifold learning can be used to detect a material morphology and structural changes over the video data. In climate science, Large Eddy Simulation (LES) and Direct Numerical Simulation (DNS) will generate exascale data from simulation outputs. Simulation output analyses on the fly can help steer simulation parameters to generate more meaningful data. In biology, clustering analysis of metagenomics data can be applicable to assembly quality improvement and abundance profile analysis, among others. General questions include Can dimensionality reduction techniques (such as manifold learning) be used to recognize actionable patterns from large volumes of high-dimensional data?

This presentation looks at approximate manifold learning algorithms such as MCFS (Multi-cluster Feature Selection) and Spectral Clustering (SC) and moves them into streaming environment to obtain better performance. Speeding up $O(N^2)$ distance calculation and $O(N^3)$ eigenvalue algorithms are addressed.

Srini Parthasarthy : Ego-net Sketching for Streaming Graph Analytics

Paper: http://streamingsystems.org/Papers/Parthasarathy.pdf
Slides: http://streamingsystems.org/Slides/parthasarathy-STREAMS16.pdf

Large-scale graph and network problems include graph sparsification, community detection, dense subgraph detection, and link prediction. Other applications include computing various measures of interest like local triangle count. Social networks generate large scale graphs, for example, Twitter generates about 500 million tweets per day, for an average of 5800 tweets per second.

Ego-net sketching keeps a L-hop (L>=1) neighborhood of each node and uses O(n) memory instead of O(e), for a graph with n nodes and e edges, which is particularly useful for graphs where e >> n. It maintains an in-memory neighbor sampled subgraph, bounded by a user configurable memory limit. Symmetrization and similarity-based techniques can recover a significant portion of the original graph. Therefore, quality of results is competitive with often significantly better computational performance than other methods. A future challenge is parallelization to use multi-core CPU, GPU and MIC architectures

12:40 - 1:00 Applications
Alex Szalay: Streaming in Astronomy
Slides: http://streamingsystems.org/Slides/szalay-stream2016.pptx (STREAM2016)
Slides: http://streamingsystems.org/Presentations/alex%20szalay%20stream-2015.pptx (STREAM2015)
Video: http://streamingsystems.org/Videos/Speaker%205%20Alex%20Szalay.mp4 (STREAM2015)

This talk describes the Sloan Digital Sky Survey SDSS, started in 1992 and finished in 2008 with 100TB of processed data. Such large data sets are just the hint of the future with more data from fewer telescopes and large simulations present additional challenges. It is important to realize that only O(NlogN) algorithms or better are realistic to process current and future data sizes. Further statistics are not really really the issue -- it's all about systematic errors. Szalay described approaches based on streaming and sampling aiming at robust techniques with a focus on dimensional reduction with a streaming PCA (Principal Component Analysis) with random projections and importance sampling. Time domain data is of growing importance and requires fast triggers. The emergence of numerical laboratories to examine results of simulations was discussed in areas like turbulence/CFD, cosmology, ocean circulation and materials science. The example of halo-finding (identification of a gravitationally bound objects) was given with the huge memory gains from use of streaming algorithms.

1:00 PM - 2:00 PM Middleware and Software Systems
Gagan Agrawal: Can Commercial Big Data Ideas Benefit Analysis of Instrument Data
Paper: http://streamingsystems.org/Papers/Agrawal.pdf
Slides: http://streamingsystems.org/Slides/streamingsystems-doe-2016.pdf

This presentation focused on analyzing simulation outputs, and not instrument data analysis. Current in situ analytics research falls into two categories: 1) application-level in situ algorithms including indexing, compression, visualization, and other analytics; and 2) system-level in-situ resource scheduling platforms that aim to enhance resource utilization and simplify the management of co-located analytics code. These in situ middleware systems target scheduling underlying tasks such as cycle stealing and asynchronous I/O. The emphasis is that there would be work on programming models for in situ algorithms. Resource scheduling can enhance the resource utilization; for example, systems such as FlexPath, DataSpaces, Glean, ADIOS, etc. achieve this. Can in situ algorithms be

seamlessly combined with systems level resource scheduling? Prior work has explored use of MapReduce in an in situ environment: which is the subject of F. Zheng, et al., PreDatA- Preparatory data analytics on peta-scale machines, http://www.cercs.gatech.edu/tech-reports/tr2010/git-cercs-10-01.pdf. The work developed in this study comprises a series of middleware systems providing MapReduce link APIs in HPC and was applied to APS data at ANL.

Andre Martin: Elastic and Secure Energy Forecasting in Cloud Environments
Paper: http://streamingsystems.org/Papers/Martin.pdf
Slides: http://streamingsystems.org/Slides/Stream-2016-Martin.pdf

This presentation focused on creating an elastic and secure processing of streaming data, based on STREAMMINE3G, an elastic event stream processing system using Intel's Safe Guard eXtensions (SGX) technology. SGX allows for a trusted enclave, where the enclave memory can NOT be accessed from non-enclave code; enclave code, on the other hand, has access to outside code and data. Intel SGX only deals with 128MB of data this is based on their page cache. A challenge here is dealing with large operator data and efficiently passing data between the two worlds (one enclave to another)?

Andre Luckow: Pilot-Streaming: Design Considerations for a HPC Stream Processing Framework
Paper: http://streamingsystems.org/Papers/Luckow.pdf
Slides: http://streamingsystems.org/Slides/20160323_PilotStream.pdf

The thought is that there is a need to couple data sources from HPC, analytics, experiments, etc. and this is a difficult problem for streaming data. The data can be extremely high velocity and we must be able to look for anomalies, for outlier detection, etc. and running on batch systems is not possible for this type of analysis. This study looked at different usage models for stream processing: coordination, real-time analytics, and analytics model update. Coordination connects a data source with the data analysis phase. Real-time analytics utilizes machine learning on incoming data for classification, etc. Analytics is where stream processing is combined with other forms of processing for real-time scoring or classification. Pilot addresses the gap in unifying streaming and batch processing for HPC applications coupling streaming data with task parallel applications.

Geoffrey Fox: WebPlotViz: Browser Visualization of High Dimensional Streaming Data with HTML5

Paper: http://streamingsystems.org/Papers/Kamburugamuve.pdf

Slides: http://streamingsystems.org/Slides/Fox-WebPlotVizSTREAM2016-Mar23-2016.pdf

High dimensional data visualization is highly valuable for scientific discovery in many fields of data mining and information retrieval. PlotViz is a 3D data point browser that visualizes large volume of 2- or 3-dimensional data as points in a virtual space on a computer screen and enable users to explore the virtual space interactively. PlotViz was initially designed to consume outputs of dimension reduction algorithms for visualizing high-dimensional data in a lower-dimensional space, such as Multi-Dimensional Scaling (MDS) and Generative Topographic Mapping (GTM). Used together with such dimension reduction algorithms, PlotViz can help users discover the intrinsic structures of high- dimensional data and browse large volumes of data points interactively and efficiently in a virtual 3D space. WebPlotViz is based on HTML5 and thus can be widely deployed. Furthermore, it supports visualization of 3D point sets (from mapping from abstract spaces) for streaming and non-streaming cases. It uses MongDB, JSON, WebGL, etc. and is open source.

Appendix D: Acknowledgements

We would like to thank the workshop attendees for valuable input and feedback during the whole process. We would like to thank Sophia Pasadis, Deneise Terry, Julie Overfield with workshop logistics. We would also like to thank Yeongshnn Ong for cover design and FLUXNET Fluxdata Team, Sean Piesert, Megha Sandesh, Nivetha Balasamy, Sean Olejaar and George Chantzialexiou, Gilberto Pastorello, Daniel Gunter with input and feedback for the report.