# Experiences Running Mixed Workloads on Cray Analytics Platforms

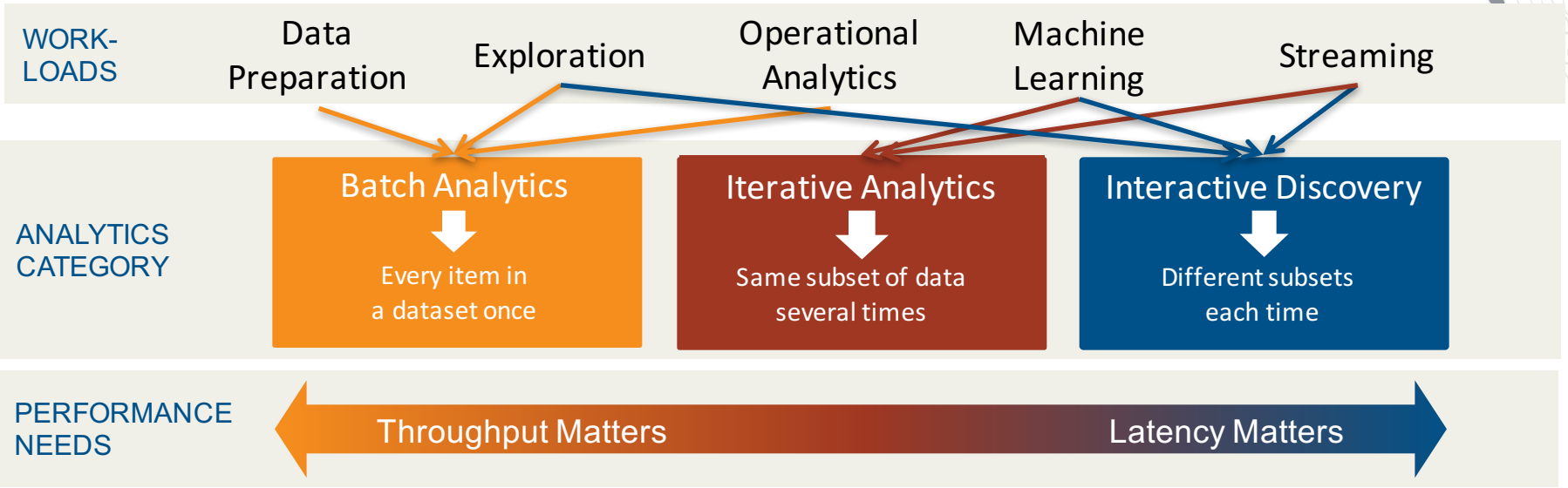
Kristi Maschhoff
Haripriya Ayyalasomayajula

# Agenda

- **Bringing HPC and Big Data together**
  - Next generation analytics platform(project name Athena)
- **Cluster Resource Management**
  - Apache Mesos
  - Spark, Hadoop and CGE frameworks on Athena
- **Social Network Analysis**
  - Description of workflow
  - Walk-through highlighting resource allocations and launch mechanisms as workflow progresses
  - Performance comparisons
- **Future Work**
- **Summary and Q&A**

# Spectrum of Analytics Workloads

## Need the Agility to Efficiently Run Mixed Workloads




**Traditional Big Data Solutions have:**

- Standalone frameworks
- Silo'd data and functions
- Costly data integration
- Poor performance at scale
- Single use

# Athena: Merging of HPC and Data Analytics

- **Integrate workloads, data sets, workflows**
  - Enable applications running in different frameworks to exchange data more seamlessly using light-weight APIs via the file system or optimally exchange data in memory
- **Performance at scale**
  - Utilize the RDMA and synchronization features of the Aries network
- **Integrated, versatile platform**
  - Dynamically supports multiple frameworks
  - Enable more complex workflows without the need to copy large amounts of data between systems

# Athena Test Platform

- **Dual-socket nodes using HSW processors**
  - 512 GB memory per node
  - 1.6 TB SSD on each node
  - Dual 1 TB HDDs per node
- **Aries-based interconnect**
- **CentOS Operating system**
- **Open Stack**
- **Mesos Resource Management Ecosystem**
- **Lustre and HDFS**

# Apache Mesos

- **Primary resource manager for the Athena platform**
- **Mesos supports a diverse set of frameworks**
  - Allocates resources dynamically to launch jobs of different frameworks
- **Mesos handles resource management**
  - Delegates scheduling decisions to frameworks
  - Can incorporate framework specific constraints such as data locality etc.
  - Bundles resources of cluster as **resource offers**
- **Frameworks can accept or reject a resource offer**
  - Accept:
    - Mesos allocates the resource
    - Framework schedules jobs on the Mesos slaves
  - Reject:
    - Wait for resource offer that satisfies its constraints

# Spark Framework

- **Apache Spark**
  - Fast, general purpose framework for large-scale data processing
- **Speed - potential to keep data in memory**
  - Solves the problem of not being able to share data across multiple map and reduce steps posed by Hadoop
  - Run programs up to 100X faster than MapReduce
- **Ease of Use**
  - Choice of language: Python, Scala, R, Java
- **Generality**
  - Supports variety of workloads within the same runtime
  - Batch, Streaming, Interactive, SQL, Machine Learning, GraphX, Complex Analytics
- **Spark runs as native Mesos framework**
  - Interacts directly with Mesos

# Building Solutions on top of Marathon

- **Hadoop and CGE are not run as native Mesos frameworks.**
  - Cray has developed solutions (interfaces to Mesos using Marathon) to support
    - Hadoop/YARN-based applications
    - Traditional Slurm- based HPC applications (initially just CGE)
      - Slurm is currently used to configure Aries communication among the sub-tasks
        - Future work to remove this additional layer
  - Marathon
    - Cluster-wide initialization and control system for running services under the Mesos ecosystem
    - Marathon registers itself as a Mesos framework
    - REST-based and provides an API for starting, stopping and scaling long-running services

# Hadoop/YARN Ecosystem

- **Built around two core components**
  - HDFS - Hadoop Distributed File system (storage)
    - Assumes fast local storage
    - Fault tolerance and resiliency ensured by replicating data across nodes
    - Data locality is offered by scheduling task execution on the nodes where the data resides
  - MapReduce (processing)
- **YARN (Yet Another Resource Negotiator)**
  - Provides resource management for workloads launched in the Hadoop Ecosystem
- **Applications in the Hadoop ecosystem access/share data using the HDFS interface**

# HDFS on the Athena platform

- **HDFS storage is supplied by the SSDs and HDDs on each of the compute nodes to form the HDFS file system**
  - Data in HDFS is accessible from everywhere and at all times
    - Includes applications running in within the Hadoop framework as well as applications running in other frameworks (Spark, CGE)
  - HDFS data stored on SSDs (on the compute node) is persistent across jobs and is globally accessible
    - Different storage model than HPC where persistent data is stored off-node in an external file system

# Differences in Hadoop and HPC environments

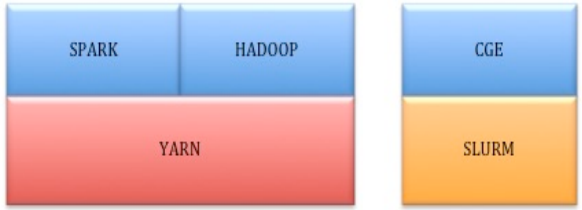
- **Hadoop projects on XC**
  - myHadoop project by Sriram Krishnan
  - CUG2014 paper by Sparks et al.
    - Discusses obstacles which complicate adoption of Hadoop as part of an workflow on an HPC system
    - Most significant – Workload management
      - Long running shared services not well suited to batch environment used for managing HPC systems
        - Database applications: Accumulo or Hbase
    - YARN resource manager/job launcher does not interoperate well with typical HPC batch schedulers

# Cray Graph Engine (CGE)

- **CGE is an in-memory Semantic Graph Database, implemented using HPC technology**
- **CGE is the follow-on to Urika-GD, using 90% of the same source code**
  - CUG 2015 talk
    - "Porting the Urika-GD graph analytic database to the XC30/40 platform"
    - PGAS programming model (Coarray C++ ) on top of DMAPP
    - Takes advantage of the RDMA and synchronization features provided by the Aries interconnect
- **CGE is based on W3C industry standards**
  - RDF graph data format (a.k.a "Triple Store")
  - SPARQL 1.1 query language
- **Cray has extended SPARQL with additional high performance graph algorithms (BGFs)**
- **CGE is designed to work with the other Athena applications to create complex workflows**

# Why multiple resource managers?

- Analytics Frameworks require a variety of resource managers
  - No one size fits all
- Hadoop Ecosystem built around YARN
- Spark can work with Mesos or YARN, and provides a standalone resource manager
- CGE and HPC applications currently uses SLURM



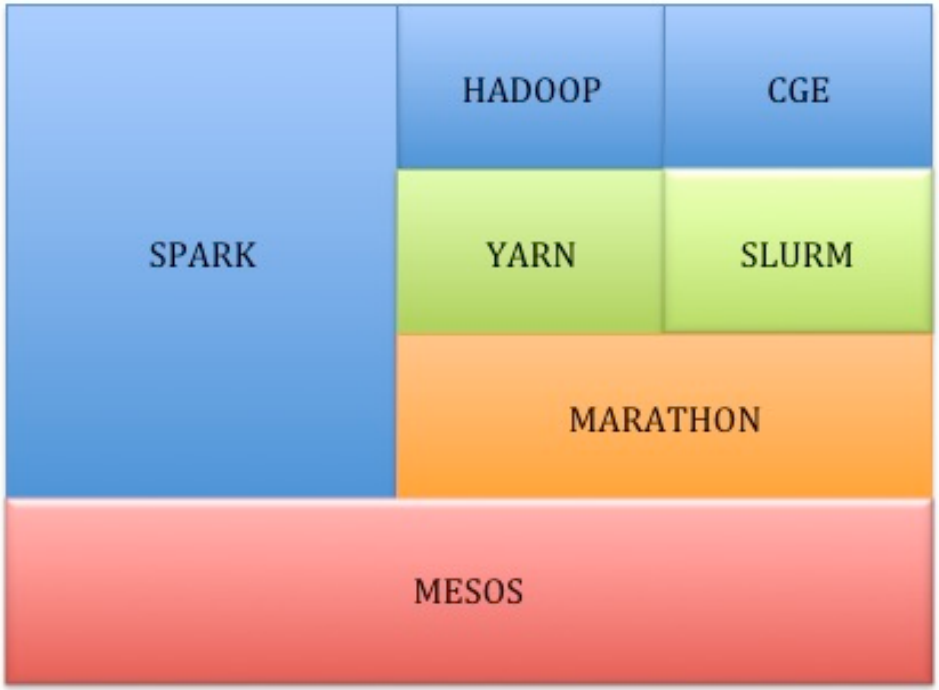
# Why not just YARN or just Slurm?

- Hadoop works with YARN only
- YARN and SLURM interoperation can be difficult
- YARN provides a scheduler integrated with the resource manager
  - Frameworks are then limited to using the YARN scheduler

On the other hand:

- Mesos has the capability to support diverse frameworks and delegates scheduling to the framework
- Selected Mesos as our prime resource manager

# Cluster Resource Management



# Social Network Analysis Workflow

- **Original workflow developed by Mike Hinchey**
  - "Implementing a social-network analytics pipe using Spark on the Urika-XA", CUG2015
  - Spark Streaming of Twitter data from Twitter4j
    - Real-time analytics pipeline
    - Batch analytics pipeline
  - Java process running on login node receives tweets as JSON records, appends to files on Lustre, and every hour file is closed, gzipped, and new file started
    - For experiments limited to 2M tweets per day
    - Full Twitter firehose is 600M tweets per day

# Social Network Analysis Workflow

- **Use Spark Streaming to process data as a series of micro batches**
  - Perform ETL ("Extract, Transform, and Load")
    - Extract needed information from each tweet
      - Parse each tweet record and reorganize data into structured data, RDDs (Dstreams) or Dataframes
        - tweets, users, relationships, hashtags
  - Real-time pipeline
    - Counting statistics, simple aggregations
  - Batch pipeline
    - Run more complex analysis, community detection

# Mixed Workloads

- **Ideal:**
  - Single framework to satisfy all data processing needs
  - Unified programming model for Spark Streaming + Spark + GraphX
- **Reality:**
  - Spark has limitations
    - Great at some tasks (ETL), not so great at others (Large-scale graph analytics)
    - Performance & Scalability
  - Managing complex workflows across multiple frameworks and programming models is challenging
    - Payback: Improved time to solution, more detailed analysis of data
- **A flexible, versatile platform makes this easier**

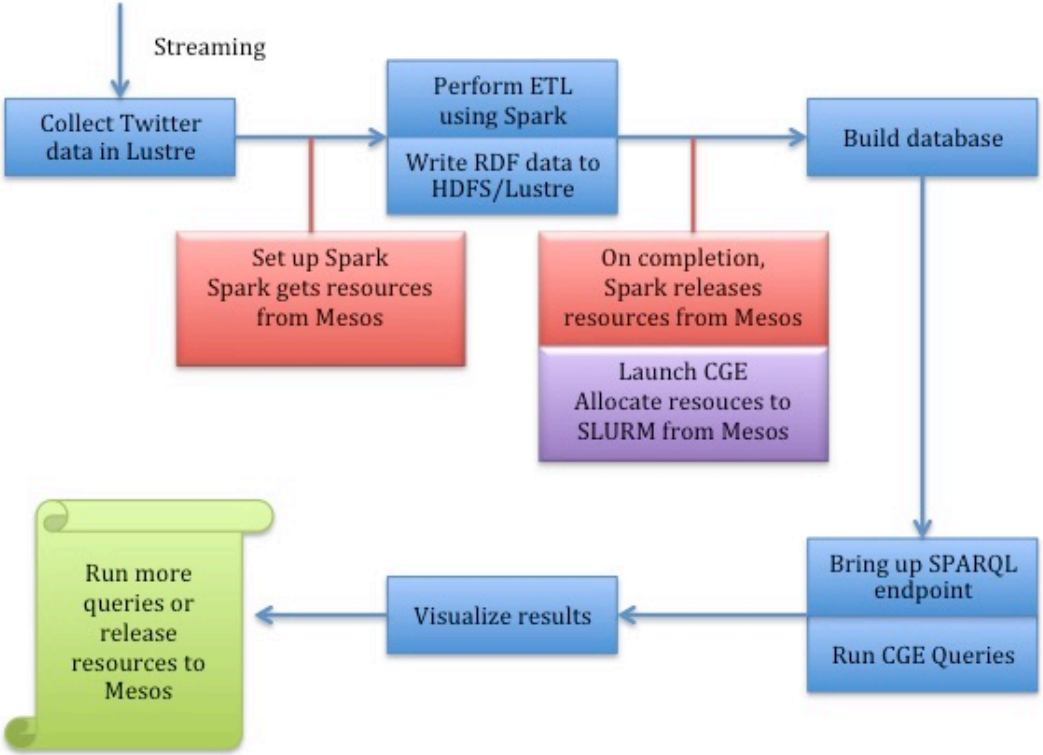
# Mixed-workload: Social Network Analysis

- **Modified Batch Pipeline: Spark + CGE**
  - Apply Spark to ETL (Extract, Transform, and Load) components of the workflow
    - Generate and write out RDF data to HDFS for later use in CGE
      - Using All_SSD storage policy for HDFS
    - Tasks at which Spark and Spark Streaming excel
  - Use CGE (SPARQL queries) to perform large joins and complex graph analysis
    - Tasks at which CGE excels

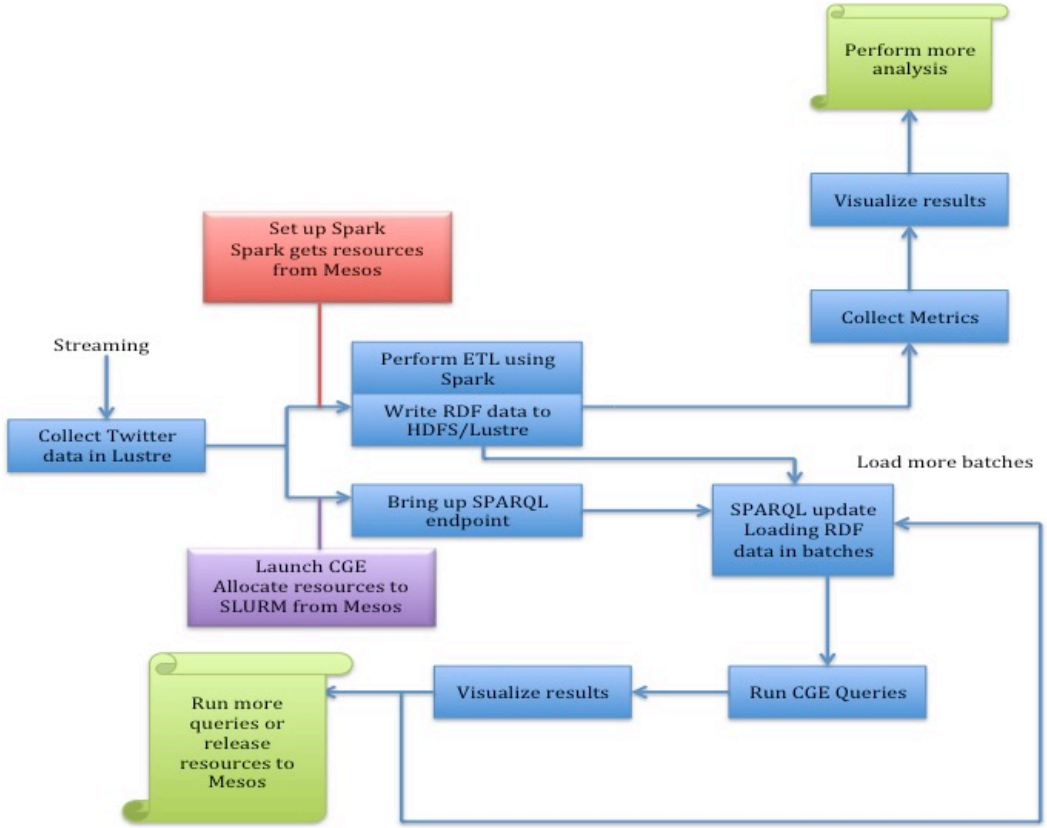
# Uber Use Case

- **Collects terabytes of event data every day from their mobile users for real-time telemetry analytics**
- **Developed a continuous ETL pipeline using Kafka, Spark Streaming, and HDFS**
  - converts the raw unstructured event data into structured data as it is collected, making it ready for further complex analytics
- **Pointer to article on Datanami**
  - http://www.datanami.com/2015/11/30/spark-streaming-what-is-it-and-whos-using-it/

# Mixed-Workload: Sequential Pipeline



# Mixed-Workload: Parallel Pipeline



# Capturing more relationships

RDF and User Network Graph Statistics

| Window | RDF Triples | Vertices | Edges | Average Edges per Vertex |
|---|---|---|---|---|
| 24 hours | 2,399,916 | 60,853 | 76,896 | 1.26 |
| 5 days | 12,033,038 | 281,536 | 404,866 | 1.44 |
| 10 days | 19,469,132 | 541,804 | 861,476 | 1.59 |
| 30 days | 54,990,937 | 1,422,267 | 2,775,008 | 1.95 |

# Mixed-workflow summary

- **Demonstrates exchange of data between Spark and CGE**
  - Using HDFS (using All_SSD storage policy)
- **Demonstrates improved time to solution**
- **Enables more detailed analysis of the data over longer periods**
  - Larger window captures more relationships between users
- **Graph analysis component of workflow (graph construction, community detection) runs 30x faster in CGE than when component is run in Spark for larger batch windows**

# Future plans

- **Investigate Apache Myriad (open-source project)**
  - Enables the co-existence of Apache Hadoop and Apache Mesos on the same analytics platform
  - Provides this ability by running Hadoop YARN as a Mesos framework
- **Working on developing a Marathon framework capable of performing the Aries setup in place of Slurm**
  - Removes a extra layer and allows more precise control of system resources
  - CGE can interface with Mesos directly, without Slurm
- **Continue to develop a more general HPC framework**
  - Bring over more of the HPC software stack running on our XC40 systems to Athena
- **Working to develop a low-level interface between CGE and Spark**

# Summary

- **Athena analytics platform**
  - Supports multiple diverse frameworks: Spark, CGE and Hadoop
  - Frameworks share platform resources dynamically
  - Solution provides an an efficient utilization of resources
- **Using a mixed-workload (Spark + CGE), we illustrated how we use Mesos to manage multiple frameworks**
  - Integrated solution enables both faster time to solution and more complex analysis of the data

# Legal Disclaimer

*Information in this document is provided in connection with Cray Inc. products. No license, express or implied, to any intellectual property rights is granted by this document.*

*Cray Inc. may make changes to specifications and product descriptions at any time, without notice.*

*All products, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.*

*Cray hardware and software products may contain design defects or errors known as errata, which may cause the product to deviate from published specifications. Current characterized errata are available on request.*

*Cray uses codenames internally to identify products that are in development and not yet publically announced for release. Customers and other third parties are not authorized by Cray Inc. to use codenames in advertising, promotion or marketing and any use of Cray Inc. internal codenames is at the sole risk of the user.*

*Performance tests and ratings are measured using specific systems and/or components and reflect the approximate performance of Cray Inc. products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.*

*The following are trademarks of Cray Inc. and are registered in the United States and other countries: CRAY and design, SONEXION, and URIKA. The following are trademarks of Cray Inc.: APPRENTICE2, CHAPEL, CLUSTER CONNECT, CRAYPAT, CRAYPORT, ECOPHLEX, LIBSCI, NODEKARE, THREADSTORM. The following system family marks, and associated model number marks, are trademarks of Cray Inc.: CS, CX, XC, XE, XK, XMT, and XT. The registered trademark LINUX is used pursuant to a sublicense from LMI, the exclusive licensee of Linus Torvalds, owner of the mark on a worldwide basis. Other trademarks used in this document are the property of their respective owners.*

# Q&A

Kristi Maschhoff
kristyn@cray.com

Haripriya Ayyalasomayajula
hayyalasom@cray.com