Pilot-Streaming: Design Considerations for a Stream Processing Framework for High-Performance Computing

Andre Luckow, Peter M. Kasson, Shantenu Jha
STREAMING 2016, 03/23/2016
RADICAL, Rutgers, http://radical.rutgers.edu
Motivation

There is a need to couple data sources, HPC, analytics! 20+ applications identified at STREAM16

Challenges:
• Data applications and pipelines are complex
• **Scalability and Elasticity**: dynamic changes in resource demands
• **Scheduling and provisioning of resources**: right amount of resources at right time
• **Programming models**: HPC (MPI, OpenMP, GPU) vs. Big Data (Java, Python, R)
• **Interoperability**: Data sources sinks often in different environments (IoT, cloud, HPC, HPDC) than compute

Current State:
• Streaming (in sciences) often implemented on application-level (w/ limited re-use)
• Manifold landscape of streaming tools (Apache Open Source Tools, Cloud Tools)
Workload Characteristics

HPC Resource

Simulation

Analysis

HPC Resource 1

Simulation

Analysis

HPC Resource 2

Simulation

Analysis
Workload Characteristics

HPC Resource 1

Simulation

HPC Resource 2

Analysis 1

HPC Resource 3

Analysis 2

Message Broker
Introduction Pilot Abstraction

User Application

Resource A
Resource B
Resource C
Resource D

Pilot-Job System
Pilot-Job
Pilot-Job
Policies

Resource Manager

System Space

User Space

User Application

Resource A
Resource B
Resource C
Resource D

http://arxiv.org/abs/1207.6644
The Convergence of HPC and “Data Intensive” Computing

A Tale of Two Data-Intensive Paradigms: Data Intensive Applications, Abstractions and Architectures In collaboration with Geoffrey Fox (Indiana) http://arxiv.org/abs/1403.1528
Pilot-Abstraction for HPC and Hadoop Interoperability

Mode I: Hadoop on HPC

- HPC Scheduler (Slurm, Torque, SGE)
- YARN/HDFS

Mode II: HPC on Hadoop

- HPC App (e.g. MPI)
- Pilot-Job
- Hadoop Application Scheduler (e.g. Spark, Tez, LLama)
- Pilot-Job

Application-level Scheduling

System-level Scheduling

Map Reduce
Spark-App
Other YARN App
Hadoop/Spark App
HPC App (e.g. MPI)

YARN
Spark
Pilot-Job

Streaming and Batch Computing

Questions:
- How to manage batch and streaming frameworks side-by-side?
- How to enable interoperability between different programming system/models/middleware/schedulers?
- How to enable elasticity?

Data

Brokers

Compute
(e.g. YARN, SLURM, Torque, PBS)

ETL
Hadoop SQL
Machine Learning

Streaming Framework

Storage and Format
(e.g. Lustre, HDFS,...)

Raw Text HDF5 Columnar Mutable/Random Access Other

Message Broker Storage Stream Processing

http://dx.doi.org/10.5281/zenodo.47946
Pilot-Streaming

Distributed Application

Pilot API

Pilot Compute
- SAGA
- Cloud
- YARN
- SSH
- iRODS
- Globus Online
- Cloud
- HDFS
- Kafka

Pilot Data

HPC
- Local/Parallel FS (SSH/GO)
- GFFS
- Node
- Pilot Agent

HTC (OSG/EGI)
- Local (iRODS)
- SRM (iRODS)
- Node
- Pilot Agent

Cloud
- Local / EBS (SSH)
- S3 (HTTP)
- EC2 VM
- Pilot Agent

Hadoop
- HDFS (WebHDFS)
- YARN
- Pilot Agent
Conclusion

1. Pilot-Jobs enable the co-location of HPC/Simulations and Big Data Tools (Hadoop, Spark, higher-level tools)

2. Pilot-Streaming will support message-broker as data source/sink that enables the de-coupling of applications

3. Dynamic resource management provided by the Pilot-Abstraction is critical for stream environments
Thank you!