Building A Data-center Scale Analytics Platform

Sriram Rao

Scientist/Manager, CISL

CISL: Cloud and Information Services Lab

- Started in May 2012
- Mission Statement: *"Applied research lab working on Systems and Machine Learning Big Data technology. Carry out innovative research by building real systems, publishing papers, and contributing to open-source."*
- CISL works closely with Microsoft Big Data teams
- CISL consists of two sub-teams
  - CISL Systems
  - CISL Data Science: Focus on ML

Data Analytics in 2010's…

- Enterprises/cloud providers are building/operating BIG clusters
  - Clusters are big…10's of 1000's nodes and cost $$$ to build and operate
- Diverse application frameworks:
  - Map-Reduce, Spark, Storm, Hbase, Giraph, …
- Leverage open source for systems infrastructure
  - Hadoop, HDFS, Linux, Puppet, Java etc.
- Key goal for datacenter operators: Maximize ROI
  - "Do more on the same hardware"
- Clusters are evolving from one per application to a shared platform in the datacenter

“Shared” Data Analytics Platform

Batch jobs

Interactive queries

Streaming queries

Production jobs

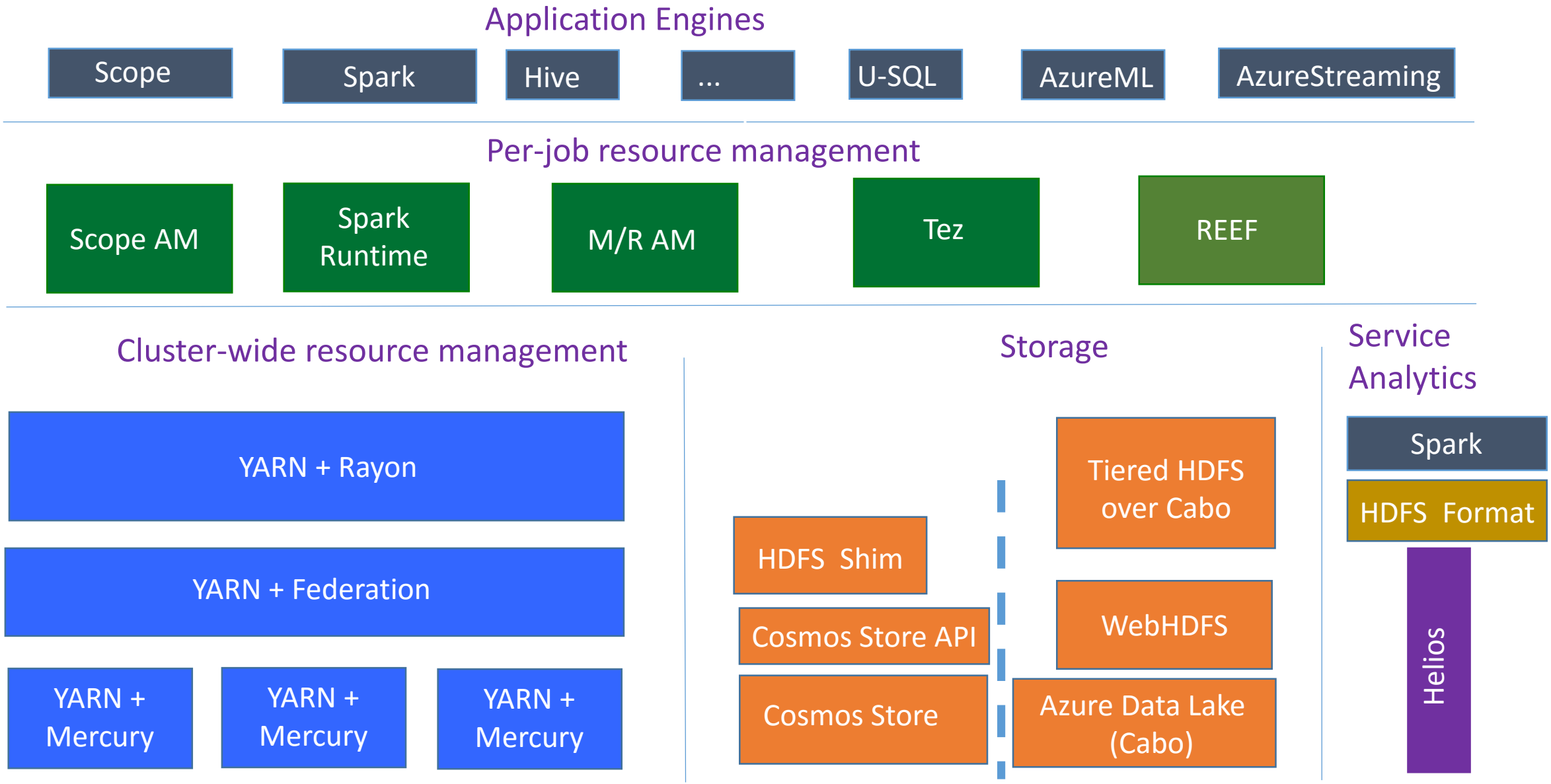
Ad-hoc jobs

Vision: One cluster to rule them all

- Can we build a scale-out analytics platform that:
  - Supports a diverse mix of applications from batch jobs to interactive queries to long-running services
  - Concurrently runs a mix of production jobs with SLAs and ad-hoc jobs
  - Maximizes cluster throughput
  - Scales to 10's of 1000's of nodes
  - Always available, software is backwards compatible, support rolling upgrade, auto reconfigure when components fail, …

Ambitious multi-person, multi-year agenda

BigData Analytics Stack



(Internal) Product Impact at Microsoft

- Much of our work influenced Microsoft's Azure Data Lake efforts
- Microsoft's clusters used for internal workloads will eventually shift to "YARN++ based" stack
- What we have done so far "lays the plumbing" for the next round of innovative research

(Selected)Publications

- Papers published in top conferences
  - Amoeba [SoCC'12]: Ganesh A., Sriram Rao, Chris Douglas, Raghu Ramakrishnan, Ion Stoica
    - Work conserving preemption to Hadoop; code ships starts with Hadoop 2.1
  - Apache YARN [SoCC'13]: Authored by CISL (Chris Douglas, Carlo Curino), Yahoo!, HW, Inmobi
    - Won Best Paper at SoCC'13
  - Rayon [SoCC'14]: Carlo Curino, Chris Douglas, Djellel Difallah, Subru Krishnan, Raghu R., Sriram Rao
    - Rayon ships starting with Hadoop 2.6
  - Tetris [SIGCOMM'14]: Robert Grandl, Aditya Akella, Ganesh A. and Srikanth Kandula, Sriram Rao
    - Enabling YARN's scheduler to pack tasks efficiently [UW-Madison, MSR, and CISL]
  - Mercury [USENIX'15]: Kostas Karanasos, Sriram Rao, Carlo Curino, Chris Douglas, Raghu Ramakrishnan
    - Hybrid scheduling combining YARN centralized with (Apollo style) distributed scheduling
    - Key components of Mercury checked in to Apache Hadoop/YARN
  - Corral [SIGCOMM'15]: Virajith Jalaparti, Ishai Menache, Peter Bodik, Sriram Rao
    - Network-aware scheduling of data parallel jobs [UIUC, MSR, and CISL]
  - Yaq [Eurosys'16]: Jeff Rasley, Kostas Karanasos, Srikanth Kandula, Rodrigo Fonseca, Sriram Rao
    - Applying network-style packet prioritization ideas to task scheduling [Brown, MSR, and CISL]

Apache YARN Overview

Background

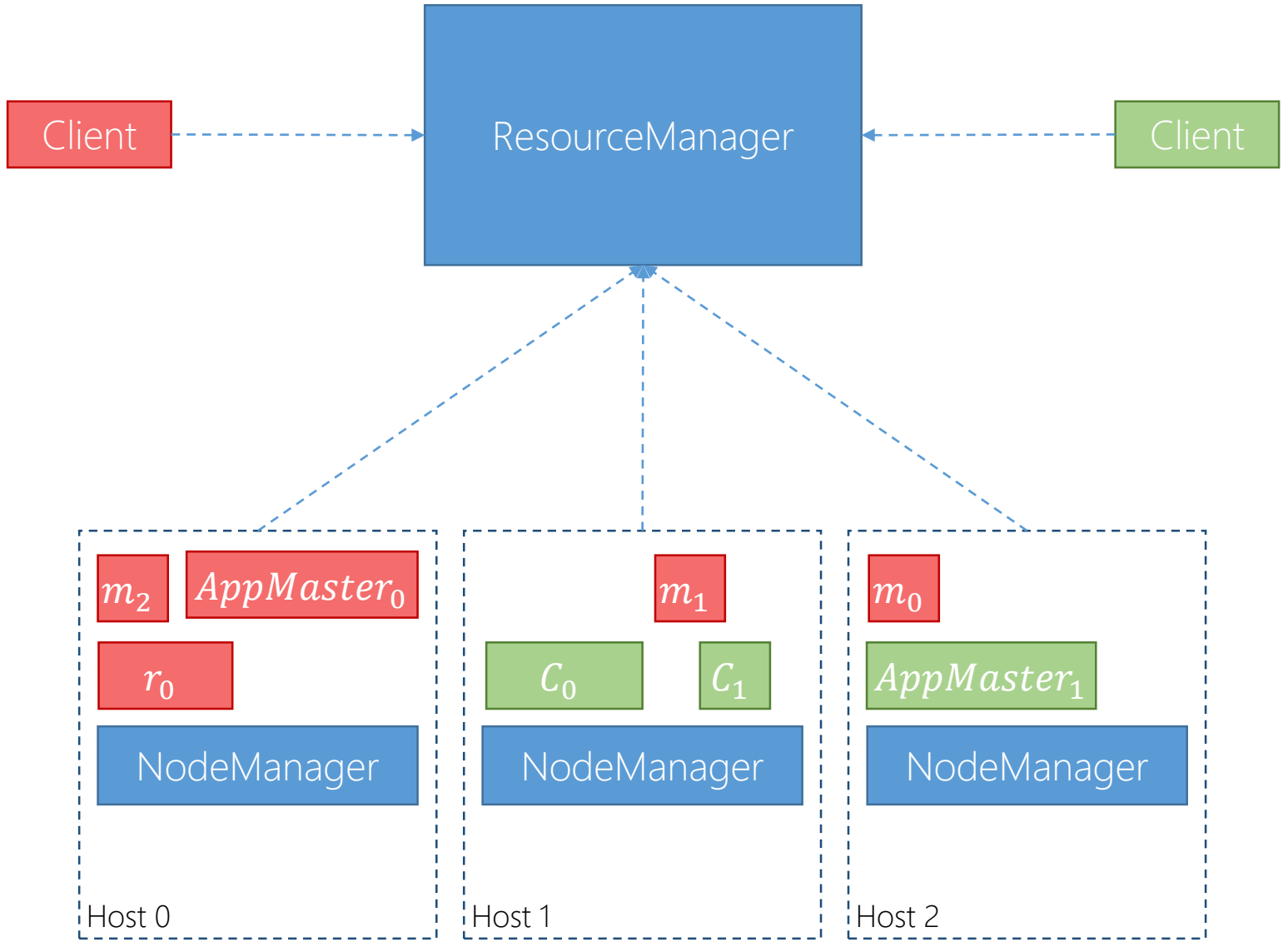
- Cluster setting:
  - BigData clusters are built using commodity hardware
  - Scale-out architecture
    - Racks of machines connected by a ToR to a central switch
- Cluster resources are exposed to applications as “containers” (aka slots)
  - Each slot is a bundle of CPU core, RAM
- For ease of management, there is a centralized scheduler that imposes policy-based sharing (capacity/fairness) of cluster resources
  - Applications negotiate with the scheduler (aka resource manager) for containers
  - When an app gets a container, it “puts” something there and runs it

Architecture

Unified resource model (elastic sharing of slots)

Heterogeneous applications

Resource vector API: declarative resource management language



Challenges with resource management

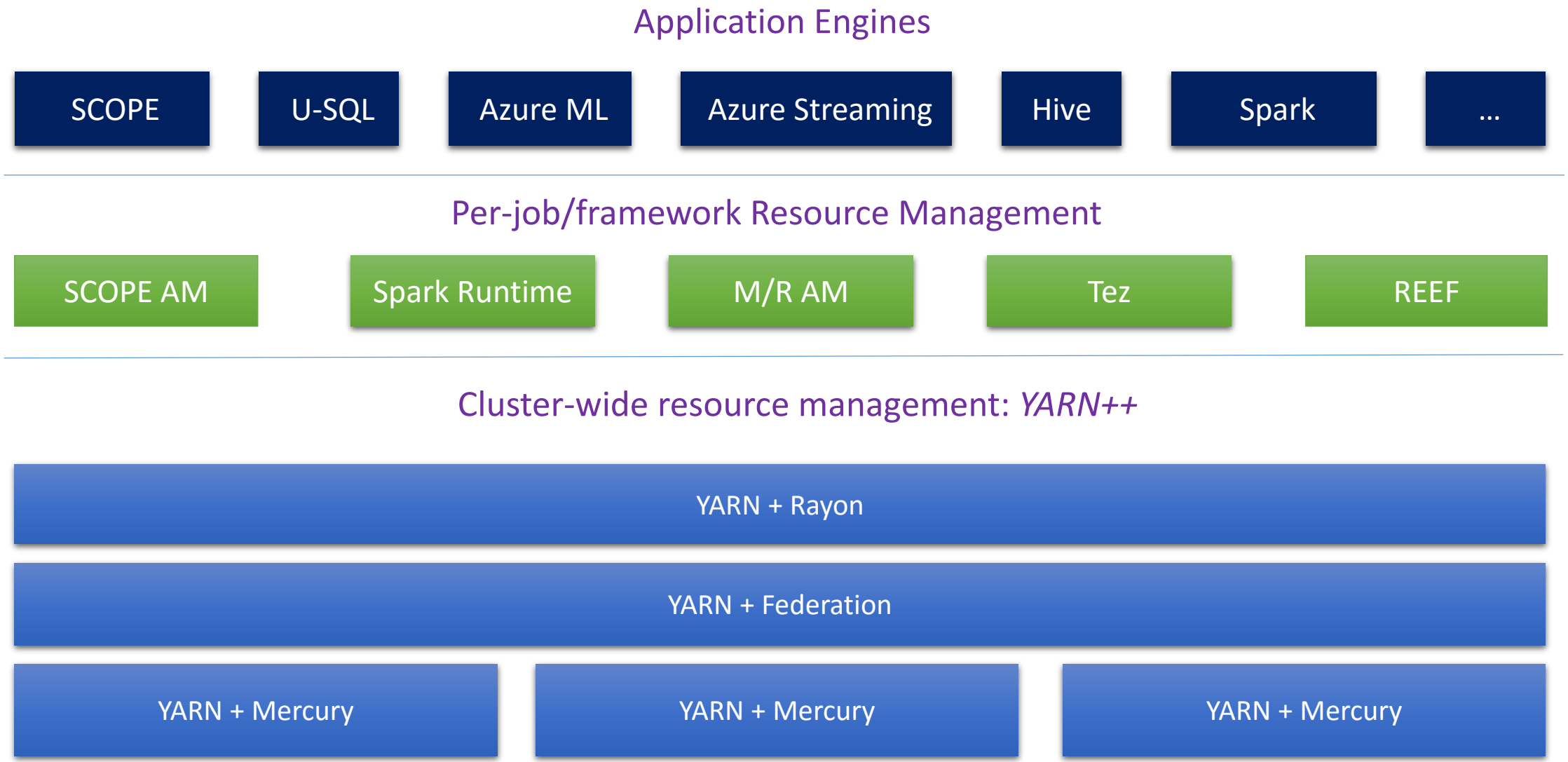
Problems

- No way to provide resource allocation SLO's to production jobs
- High allocation latency, which affects "small" jobs that are majority
- YARN RM has scalability limitations
  - Known to scale to 4K nodes

What have we done so far?

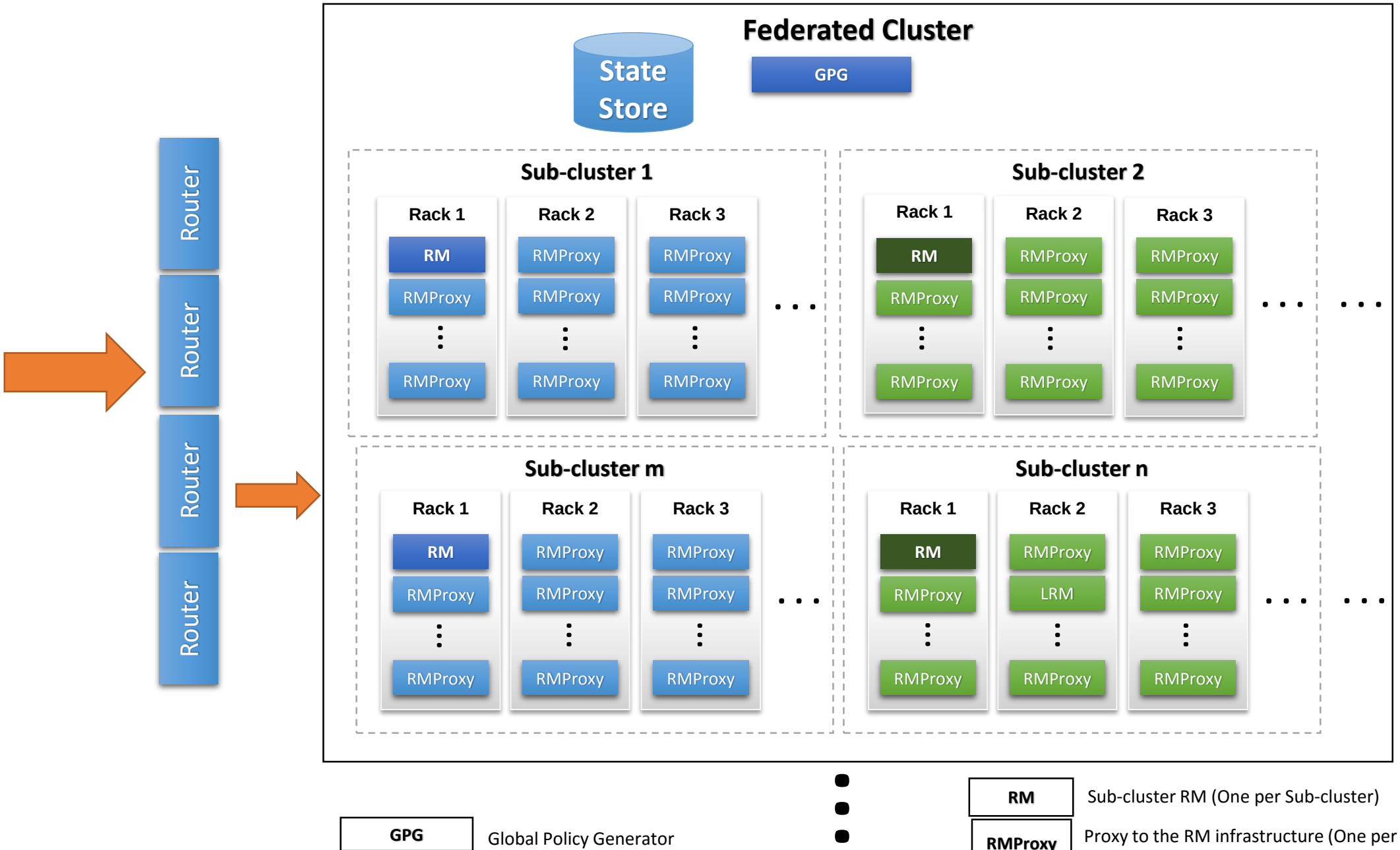
- Rayon: Resource reservation framework on YARN
- Mercury, a hybrid resource management framework that combines centralized and distributed scheduling
- Federation: Extend Mercury to build "federated" YARN clusters

Big Picture


Application Engines
SCOPE
U-SQL
Azure ML
Azure Streaming
Hive
Spark
…
Per-job/framework Resource Management
SCOPE AM
Spark Runtime
M/R AM
Tez
REEF
Cluster-wide resource management: *YARN++*
YARN + Rayon
YARN + Federation
YARN + Mercury
YARN + Mercury
YARN + Mercury

YARN + Federation (see YARN-2915)

- Based on deployments at Yahoo, LinkedIn, etc., YARN RM is known to scale to 4K nodes
  - They have multiple 4K clusters, each of which is an “island”
  - Operated in this manner for multiple years…
- How do we do resource management at datacenter scale?
  - Want to push the limits of scale ☺
- Key idea: Federation
  - 4K size sub-clusters are “bricks”
  - Federate bricks/sub-clusters to handle large sizes
    - Allows for “always on”
    - Test/flight at scale
    - Self-configuring


Federated Cluster
State Store
GPG
Router
Router
Router
Router
Sub-cluster 1
Sub-cluster 2
Sub-cluster m
Sub-cluster n
Rack 1
Rack 2
Rack 3
RM
LRM
RMProxy
GPG
Global Policy Generator
RM
Sub-cluster RM (One per Sub-cluster)
RMProxy
Proxy to the RM infrastructure (One per Node/Rack, this is an NM service)

What next on Federation?

- What we have done so far is put down a baseline architecture
  - Scale-out system, self-configuring/self-healing capabilities
  - We can leverage all the work done in OSS for YARN related improvements
- Implementation Status:
  - There is a branch in Apache for Federation
  - If you are interested in the code, reach out to me
- Need to define policies:
  - binding tenants to sub-clusters,
  - take workload characteristics in placement,
  - …

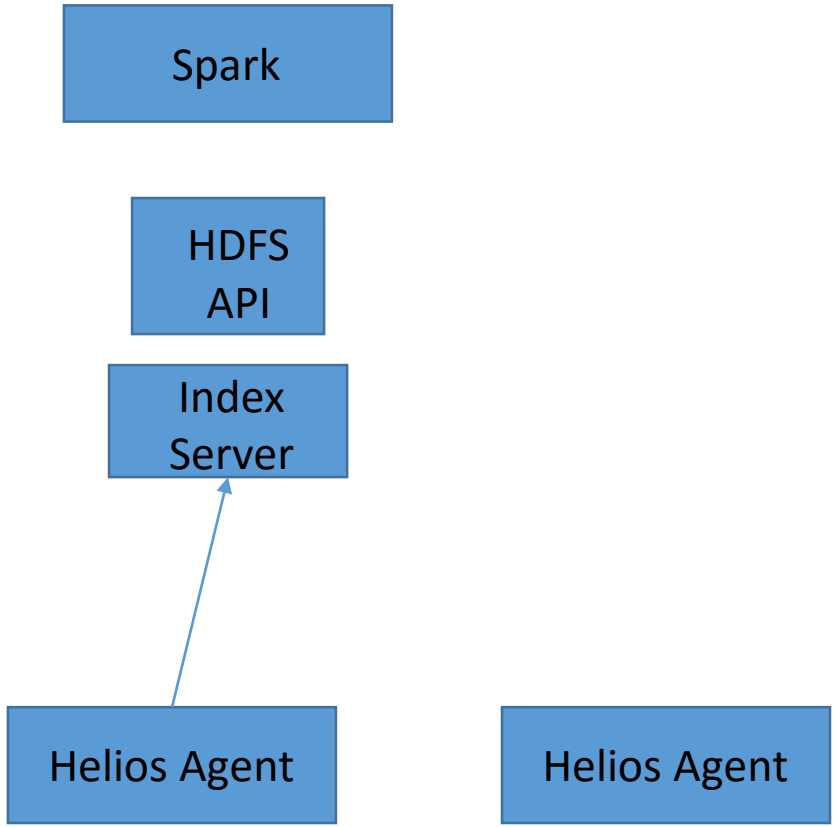
Service Analytics

Problem Setting

- For cluster operators *log data* is key to deriving operational insights
- Datacenter size clusters generate massive log data. Suppose:
  - 1 machine generates 10MB/min
  - 60k machines generate: 60k * 10MB * 60 * 24 ≈ 1PB/day!
- Want data to be queryable “near real-time”
  - This makes it possible to take corrective action should go awry
- What we realized…
  - There is no subsystem that is designed for this scale ☺

What did we do…

- Built Helios, a scale-out log collection framework
  - Agent on each machine parses/indexes the data
  - Collect at aggregation nodes, which merge indexes and serve queries
- Added a “HDFS head” at the aggregation nodes
- Leverage open-source tools such as Spark to query the log data
- End-to-end time: From the time log line is generated to when it is queryable is: **90 secs**
  - Not bad ☺



Summary

- I only gave you a flavor for the problems we are looking into
- We are a small team
- What we have is a multi-person, multi-year agenda
- We collaborate extensively with MSR, University folks

We are hiring!

Interested in summer internships, full-time researcher positions, talk to us