Scaling LinkedIn's Hadoop YARN cluster beyond 10,000 nodes

By Keqiu Hu, Jonathan Hung

LinkedIn

Speakers

Keqiu Hu

Jonathan Hung

Agenda

1 **Hadoop at LinkedIn**

2 Optimizing YARN

3 Scaling Horizontally

4 Q&A

LinkedIn's Vision

Create economic opportunity for every member of the global workforce

850M Members (July 2022)

59M Companies (Apr 2022)

52M/week Job Seekers (Jun 2022)

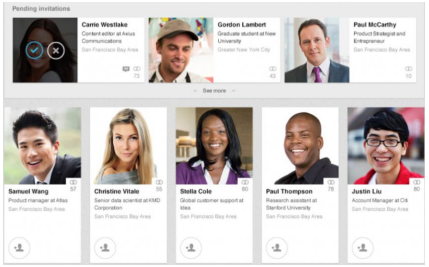
39K Skills (July 2022)

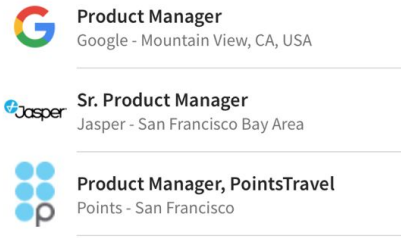
128K Schools (July 2022)

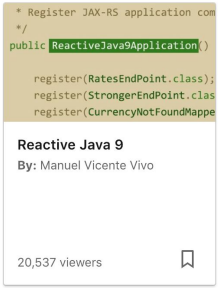
Hadoop at LinkedIn

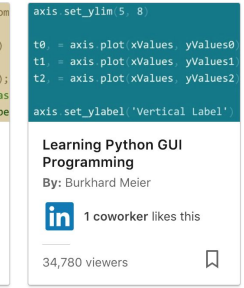
Powers:

- Data Analytics
  - Economic graphs
  - Data driven decisions
- AI
  - People you may know
  - Jobs you may be interested in
  - Courses you may be interested in
  - Feed recommendations
  - …

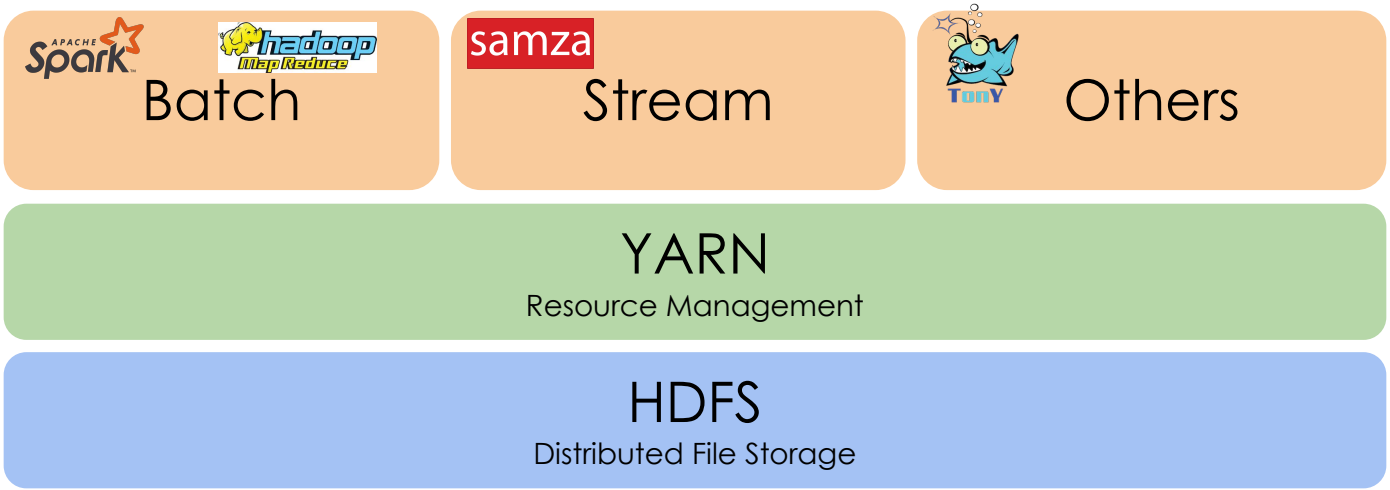




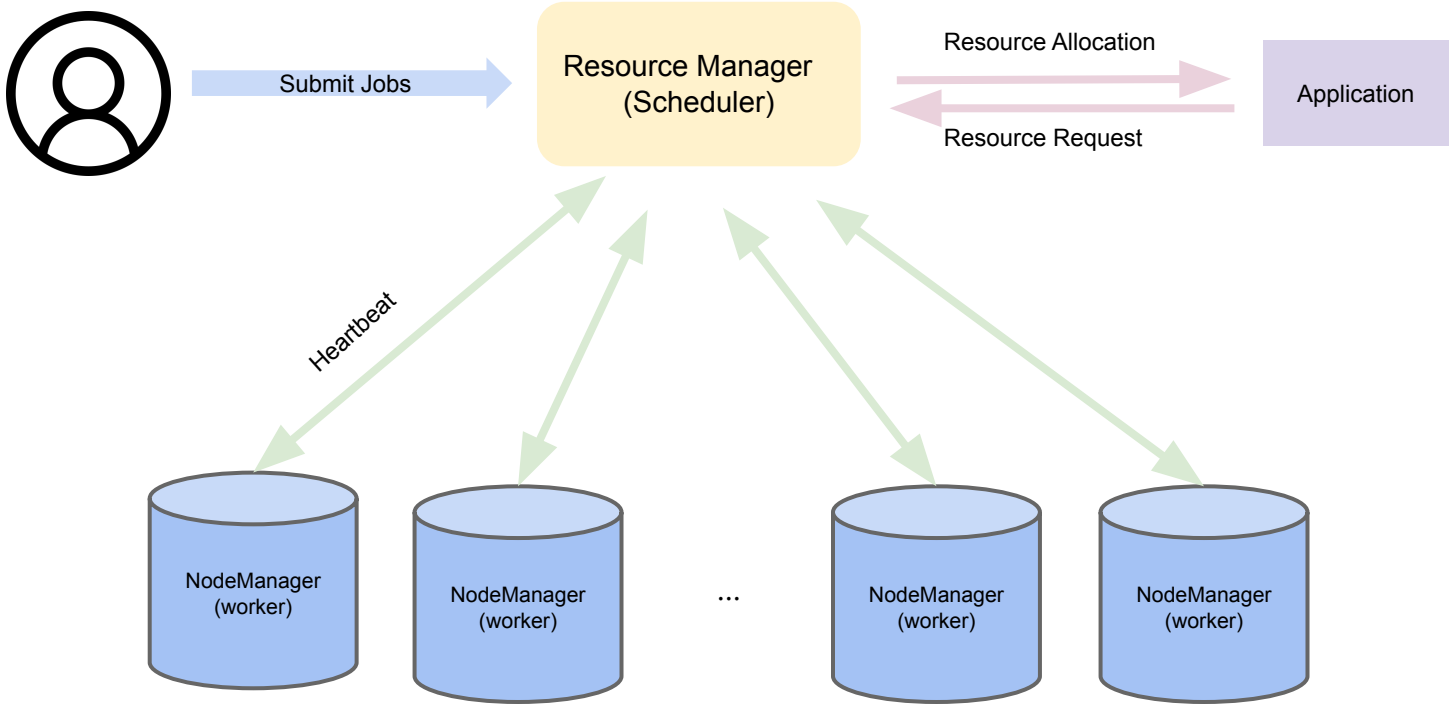




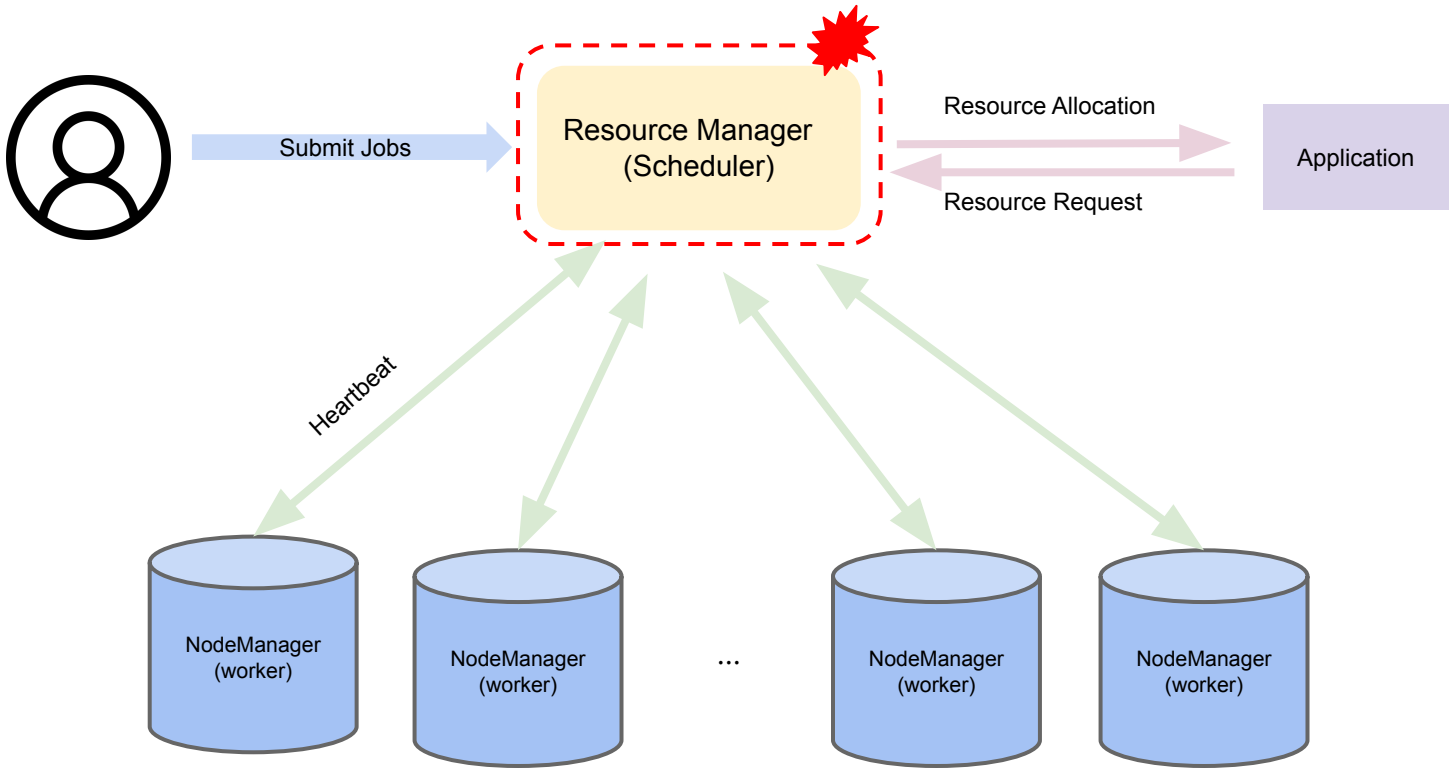
Hadoop YARN


Batch
Stream
Others
YARN
Resource Management
HDFS
Distributed File Storage

YARN Architecture (simplified)



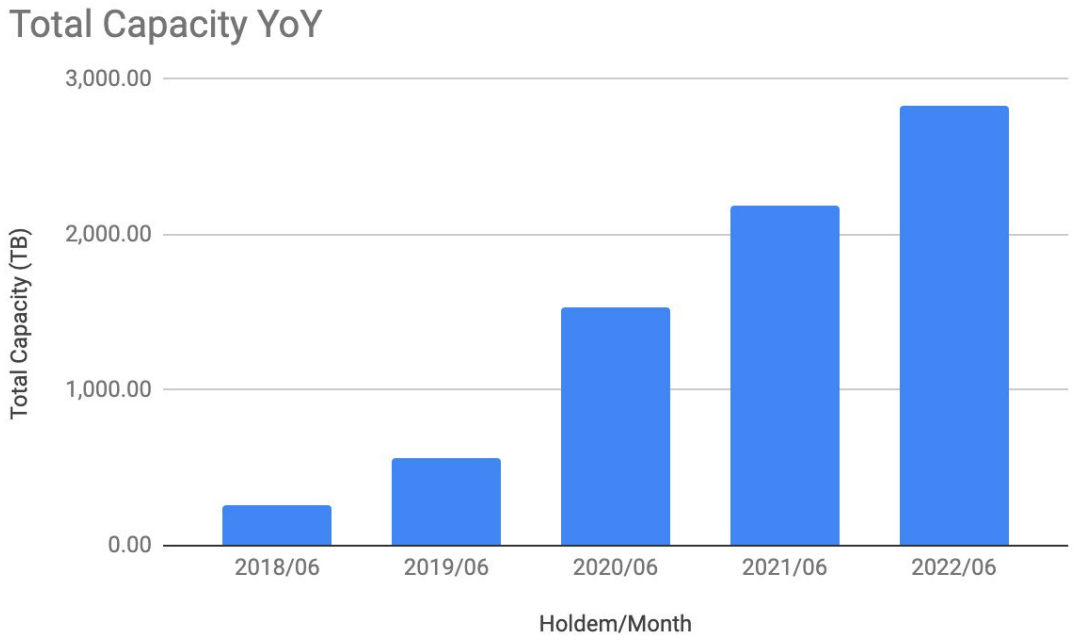
YARN Architecture (simplified)



Hadoop at LinkedIn

LinkedIn's largest cluster:

- **10x** growth over 4 years
- 15k nodes
- 3PB memory
- 1 exabyte data



Agenda

1 Hadoop at LinkedIn

2 **Optimizing YARN**

3 Scaling Horizontally

4 Q&A

Optimizing ResourceManager: Part 1

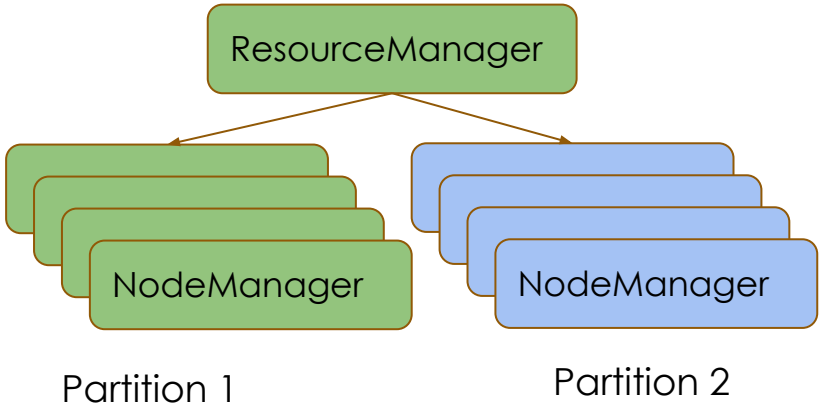
- Grew our primary cluster to 6000 nodes
- Workload grew to 300k applications/day
- Result: allocation speed dropped from 500 containers/sec to 50 containers/sec

Optimizing ResourceManager: Part 1

- Caused by inefficiencies in YARN partitioning
- Partition 2 node heartbeats to RM -> N failed attempts to schedule partition 1 applications on this node
- NMs are heartbeating every second, so each heartbeat will incur failed attempts

Pending apps:

- Partition1_1
- Partition1_2
- …
- Partition1_N
- Partition2_1
- Partition2_2
- …
- Partition2_M

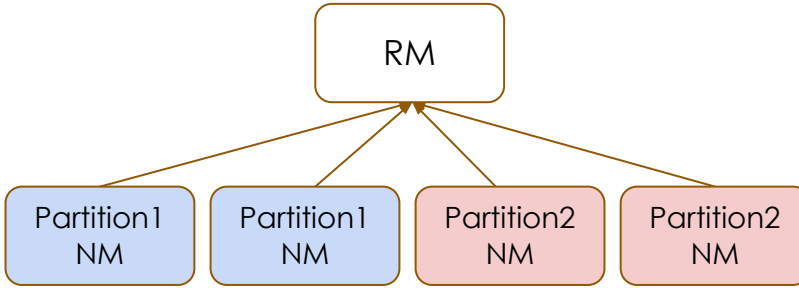


Optimizing ResourceManager: Part 1

- Solution: If partition 1 node heartbeats, only look at Partition1_1, Partition1_2, …, vice versa for partition2 nodes
- Identical scheduling behavior to having separate physical clusters, with the flexibility of having a single cluster

Pending apps:

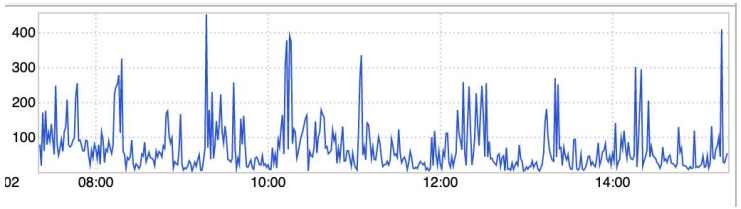
- Partition1_1
- Partition1_2
- …
- Partition1_N
- Partition2_1
- Partition2_2
- …
- Partition2_M



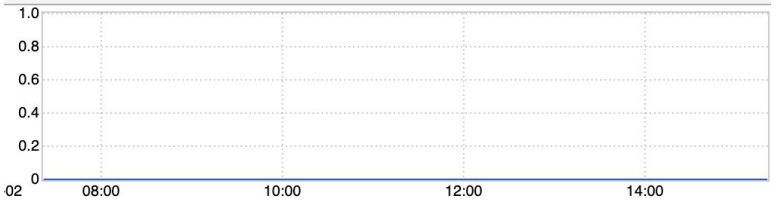
Optimizing ResourceManager: Part 2

- Overall container throughput recovered to 600 containers/sec
- Some queues were ~200 containers/sec, others 0 containers/sec
- Issue: overall cluster was running fine but some queues were running in degraded state (starvation)

Queue A



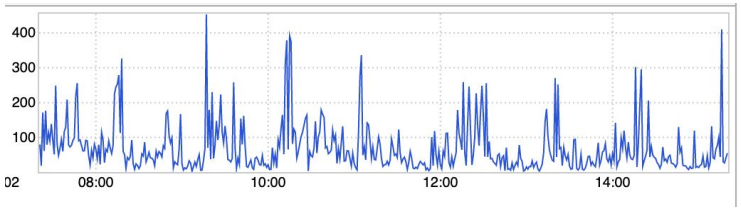
Queue B



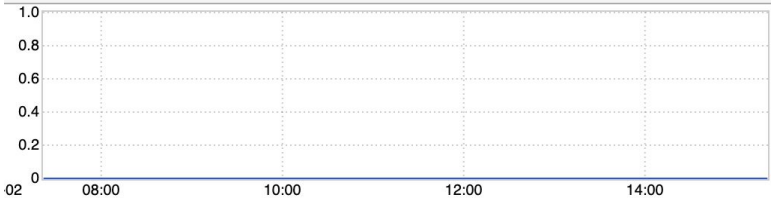
Optimizing ResourceManager: Part 2

- Issue: Large queue A < 15% capacity, small queue B > 15% capacity
- Short-lived queue A containers means high churn, scheduler cannot allocate containers faster than queue A releases => queue A remains < 15%
- Queues allocated based on capacity
  - Queue B never receives resources
- Fix: Change queue ordering policy to round robin

Queue A

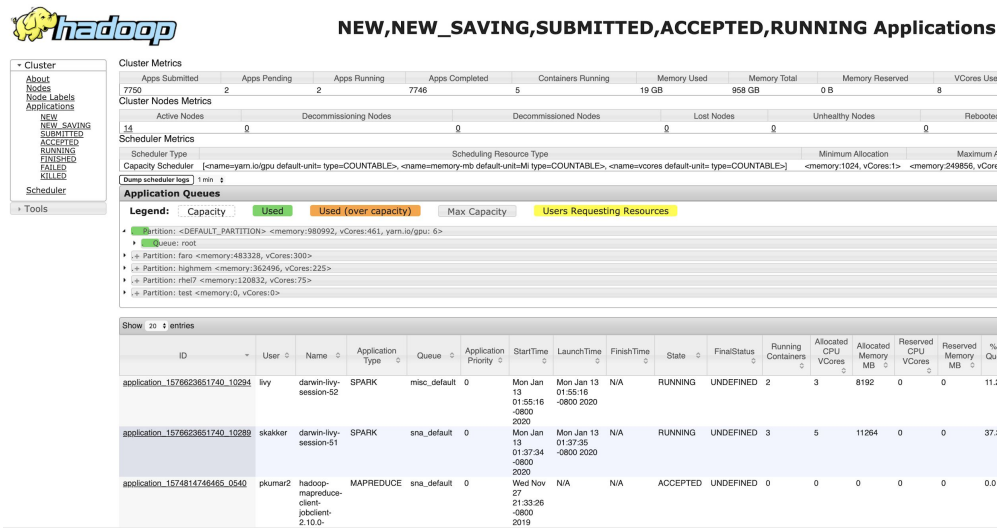


Queue B



Replaying Scheduler Activity At Scale

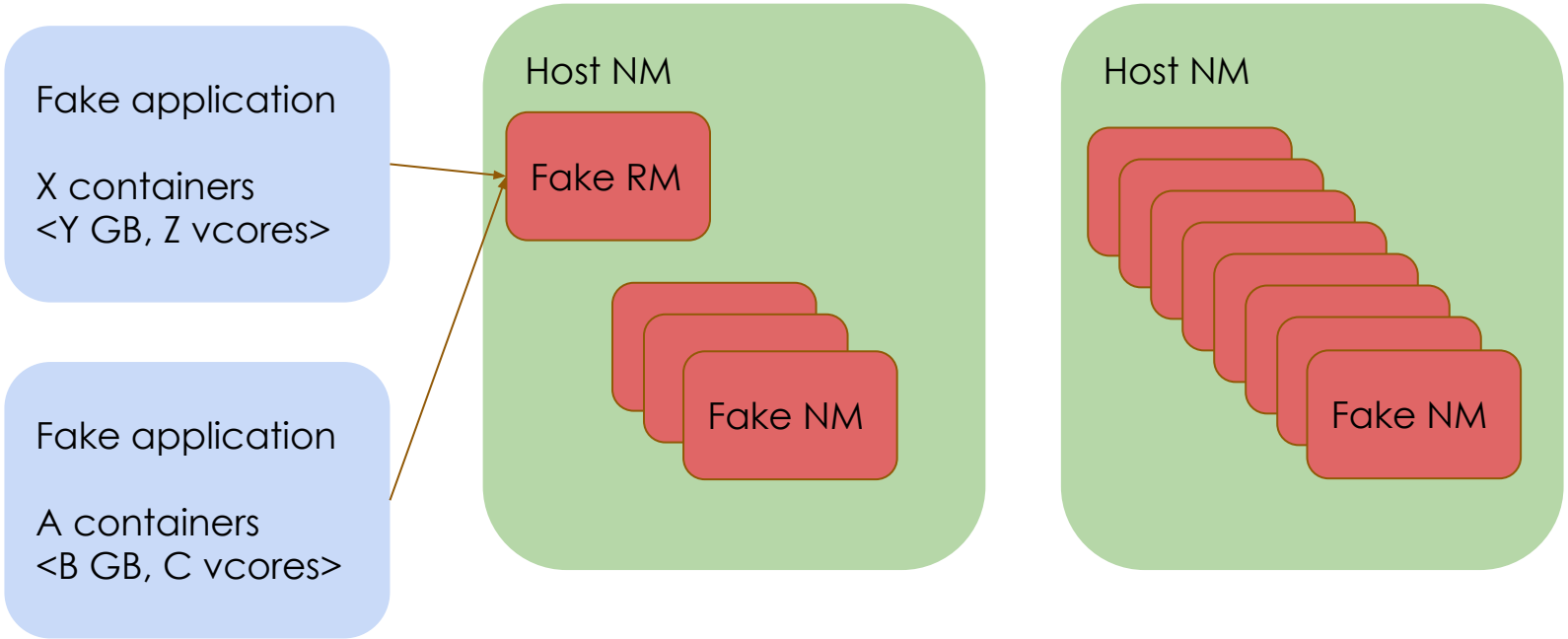
- Test clusters don't generate enough load
- Attaching profilers, enabling debug logging, etc. drastically impact scheduler performance
- How to test changes in scheduling logic under stress?



DynoYARN: YARN Scale Testing Tool

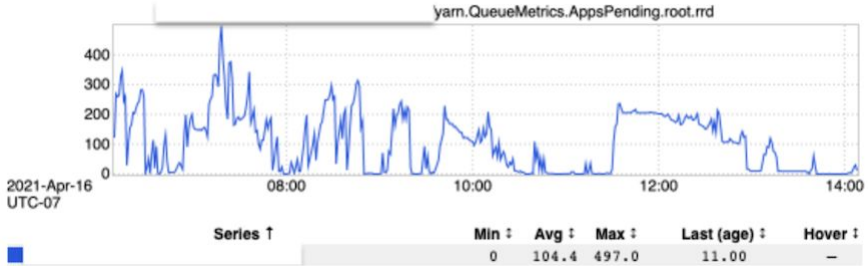
- Testing tool developed at LinkedIn to simulate large clusters and cluster load
- On-demand YARN cluster with small hardware footprint
- Simulate YARN daemons and YARN applications, without running any actual compute workload

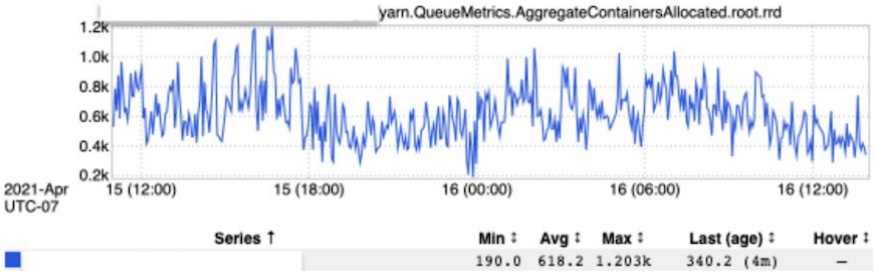
DynoYARN: YARN Scale Testing Tool



DynoYARN: YARN Scale Testing Tool

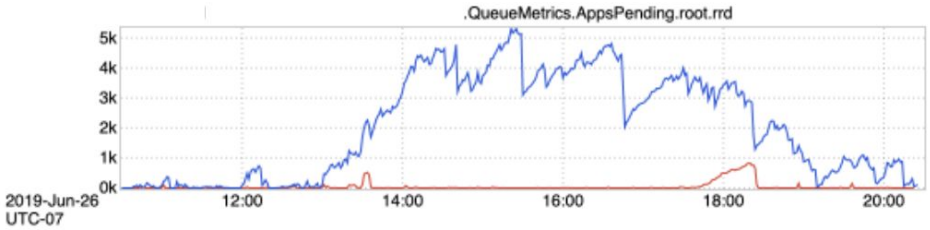
- Fake NodeManager runs in a container: 50 containers/host = 50 NM/host
  - 200 node physical cluster can simulate a 10k node cluster
- Take traces from production cluster, replay them on DynoYARN cluster
- Scale up production workload by 1.5x, 2x, gather metrics

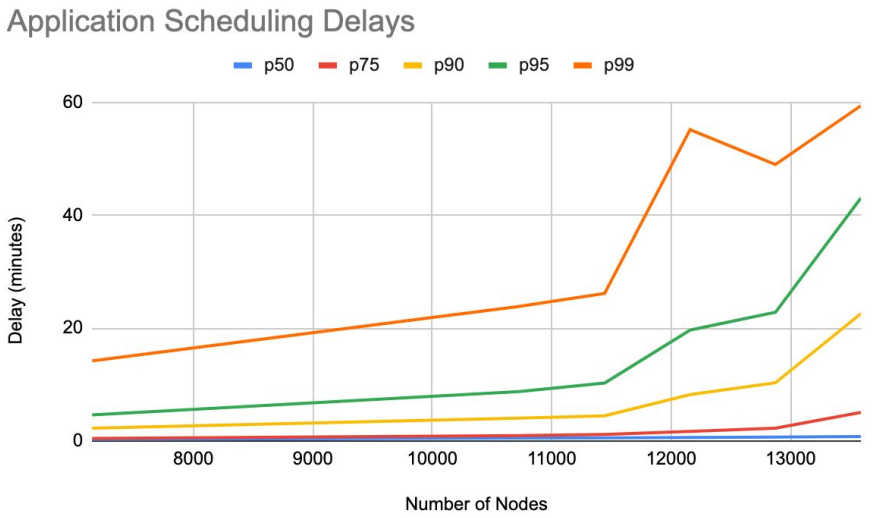




DynoYARN: YARN Scale Testing Tool

- Compare metrics with baseline and scaled production traces
- Forecast application delays with scaled traces





DynoYARN: Open Source

- Open sourced at http://github.com/linkedin/dynoyarn

Agenda

1 Hadoop at LinkedIn

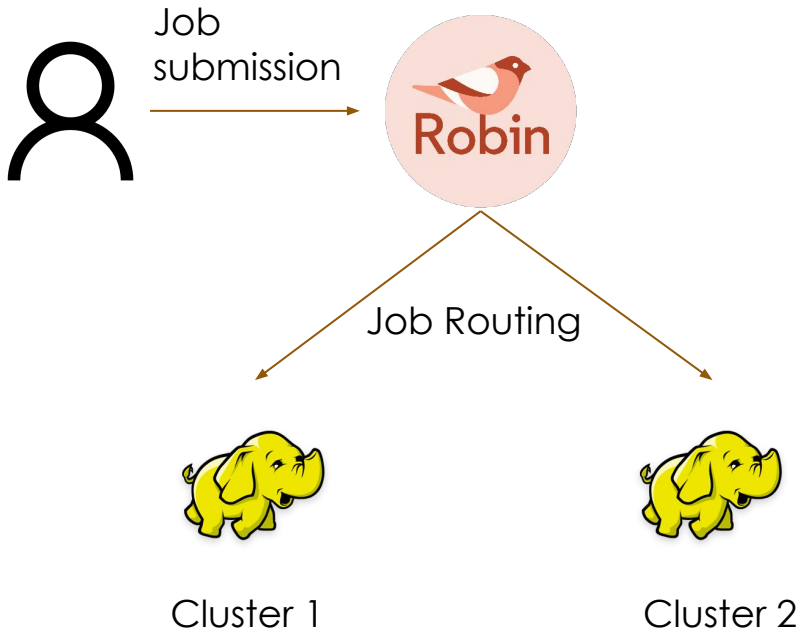
2 Optimizing YARN

3 **Scaling Horizontally**

4 Q&A

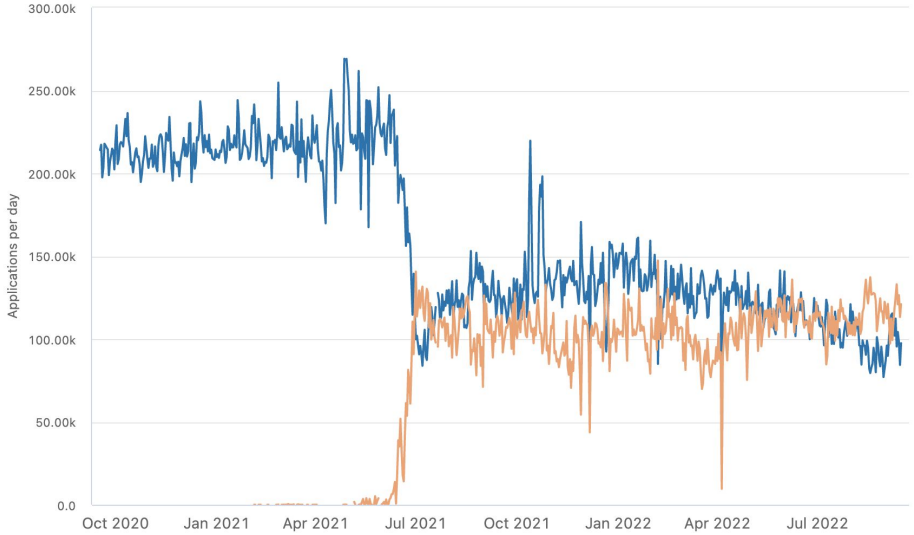
Horizontal Scaling: Robin

- YARN cannot scale indefinitely
- Fragment large YARN cluster into multiple smaller YARN clusters
- Abstract out clusters on client side so client still sees a single cluster



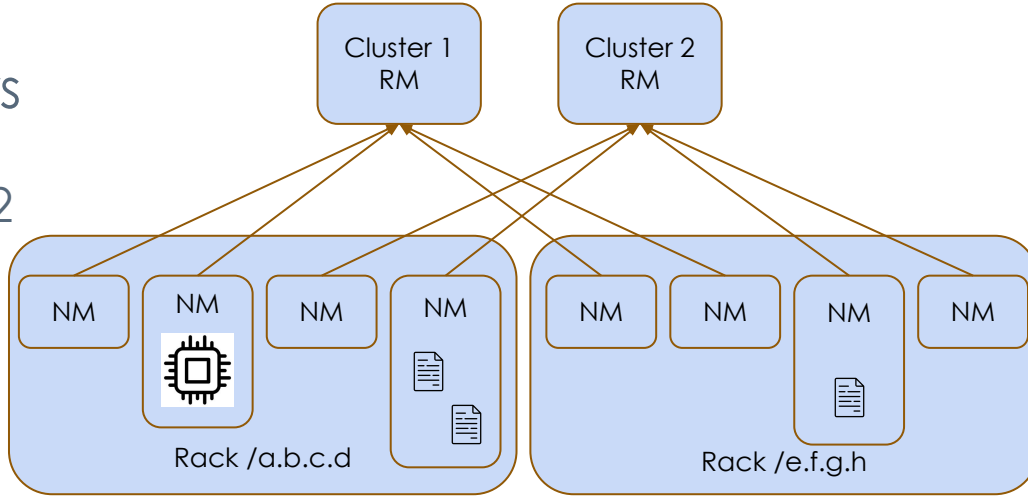
Robin

- Split 10k node cluster into two 5k node clusters
- Transparently route jobs based on cluster load



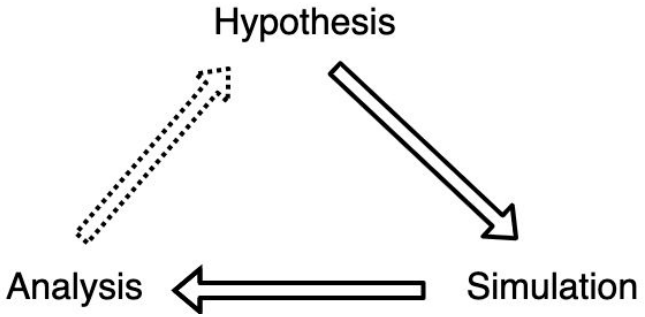
Robin: Rack Striping

- How to split NodeManagers between clusters?
  - Moving full racks to cluster 2 may result in loss of data locality (three replicas may be on cluster 1)
- Split each rack's nodes across compute clusters to guarantee rack locality



Lessons Learned

- Follow a scientific approach to testing changes
- Know your system
  - Identify meaningful metrics
  - Come up with hypotheses
- Use tools to analyze system performance



Agenda

1 Hadoop at LinkedIn

2 Optimizing YARN

3 Scaling Horizontally

4 **Q&A**