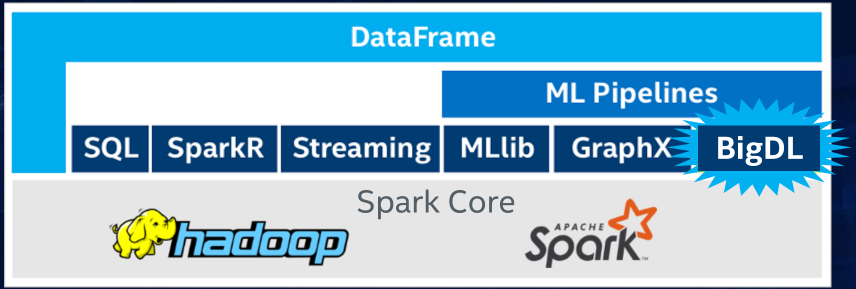
# ACCELERATING DEEP LEARNING ON APACHE SPARK USING BIGDL WITH COARSE-GRAINED SCHEDULING

Shivaram Venkataraman (Microsoft Research, UC Berkeley)
Ding Ding (Intel)
Sergey Ermolin (Intel)

June 2018

# BigDL

## HIGH PERFORMANCE DEEP LEARNING FOR APACHE SPARK* ON CPU INFRASTRUCTURE




BigDL is an **open-source** distributed deep learning library for Apache Spark* that can run directly on top of existing Spark or Apache Hadoop* clusters

**Ideal for DL Models TRAINING and INFERENCE**

**Designed and Optimized for Intel® Xeon®**

*No need to deploy costly accelerators, duplicate data, or suffer through scaling headaches!*

**Feature Parity & Model Exchange** with TensorFlow*, Caffe*, Keras, Torch*

**Lower TCO and improved ease of use** with existing infrastructure

Deep Learning on Big Data Platform, Enabling **Efficient Scale-Out**

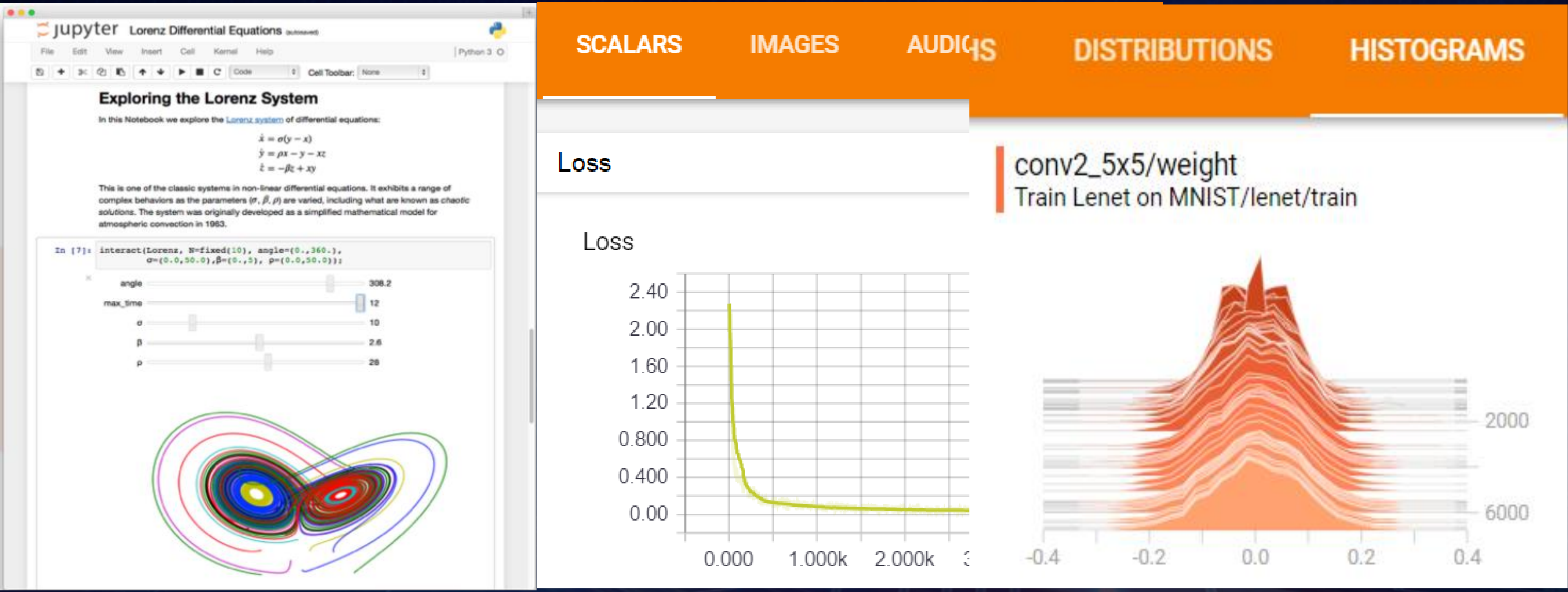
***Powered by Intel® MKL and multi-threaded programming***

bigdl-project.github.io

software.intel.com/bigdl

# BIGDL

## Jupyter, Zeppelin notebooks and TensorBoard support




bigdl-project.github.io

software.intel.com/bigdl

BigDL

# BUILDING & DEPLOYING WITH BIGDL

## PLATFORMS

bluedata
cloudera
INSIGHTEDGE
databricks
CRAY THE SUPERCOMPUTER COMPANY
DELL
KINGSOFT
Qubole
Lightbend

## CLOUD SERVICE PROVIDERS

Alibaba Cloud aliyun.com
aws
Azure
IBM Cloud
Google Cloud Platform

## SOLUTIONS

JD.COM 京东
UnionPay 银联
GIGASPACES innovate with confidence
UCSF
MLS LISTINGS

Open Source Community support:
2496 STARS | 500+ FORKS | 50 CONTRIBUTORS

https://bigdl-project.github.io

And Many More...



intel

# ANALYTICS ZOO

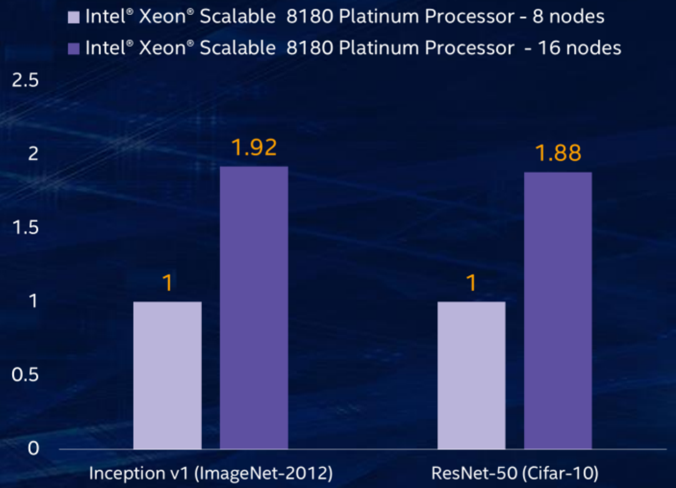
## Analytics + AI Pipelines for Spark and BigDL

**“Out-of-the-box” ready for use**

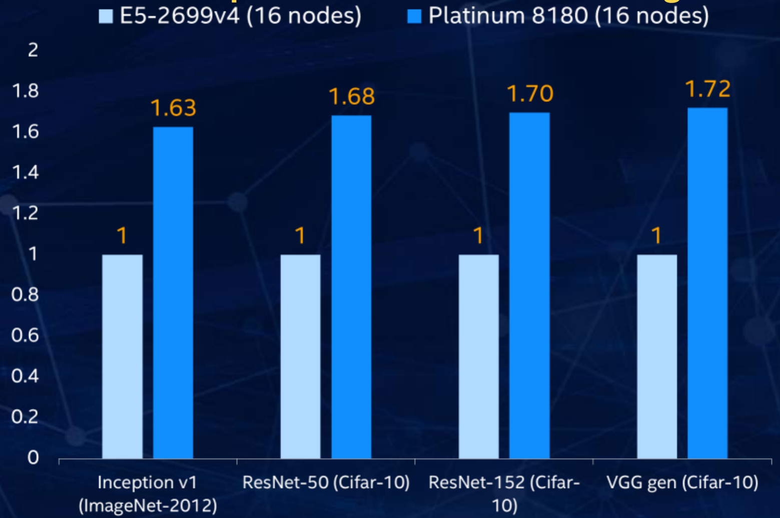
- Reference use cases
  - Fraud detection, time series prediction, sentiment analysis, chatbot, etc.
- Predefined models
  - Object detection, image classification, text classification, recommendations, etc.
- Feature transformations
  - Vision, text, 3D imaging, etc.
- High level APIs
  - DataFrames, ML Pipelines, Keras/Keras2, etc.

# DEEP LEARNING WITH BIGDL/SPARK

## Node Scaling with BigDL




## Generational performance increase with BigDL




## GET EXCELLENT MULTI-NODE SCALING AND GENERATIONAL PERFORMANCE WITH YOUR EXISTING HARDWARE

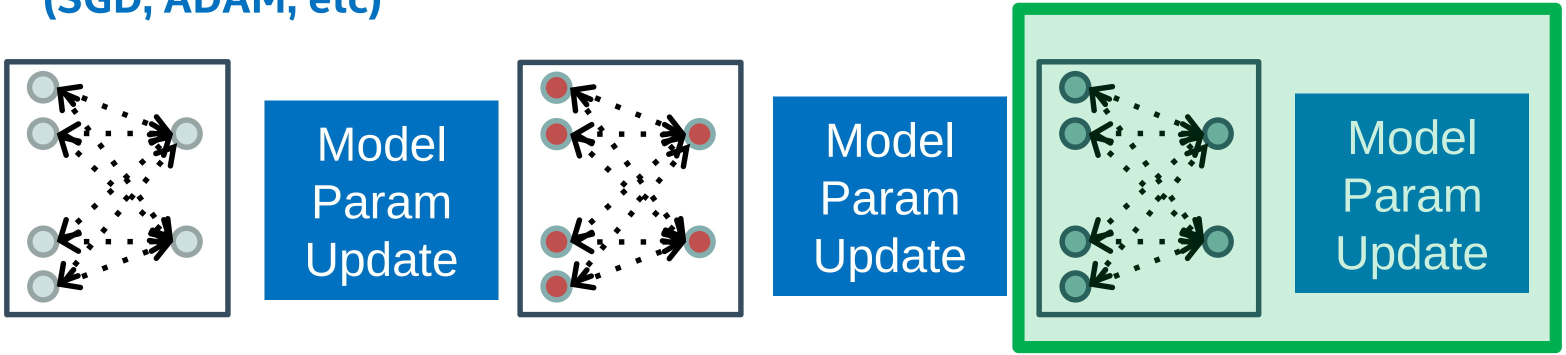


bigdl-project.github.io

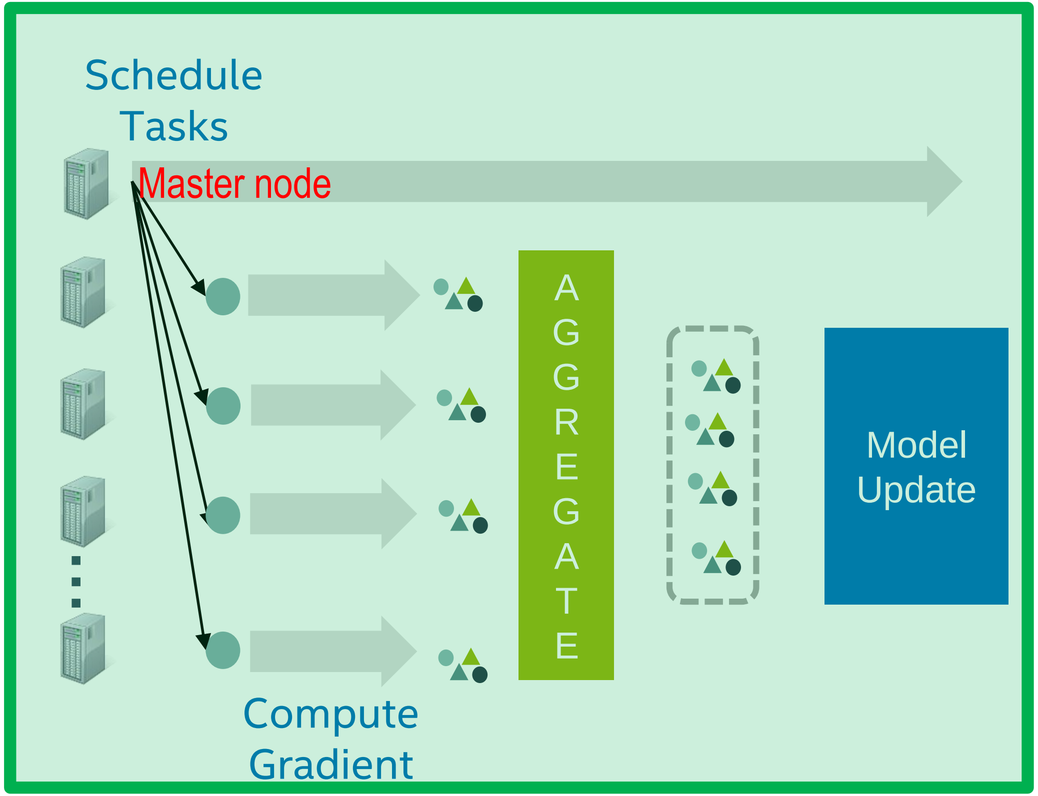
software.intel.com/bigdl

# DEEP LEARNING TRAINING

**All Iterative ML Algorithms exchange model parameters after each iterations (SGD, ADAM, etc)**



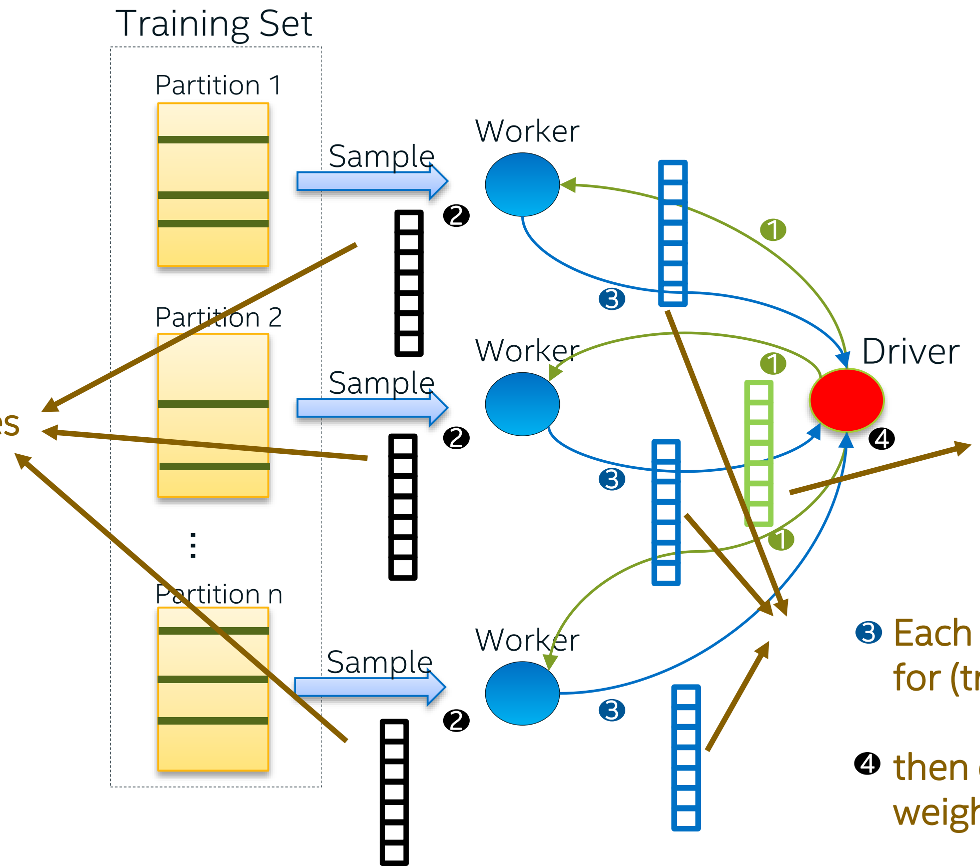
# ZOOMING IN: INSIDE EACH ITERATION




**Training**

```
for (i <- 1 to N) {
  batch = next_batch()
  output = model.forward(batch.input)
  loss = criterion.forward(output, batch.target)
  error = criterion.backward(output,
batch.target)
  model.backward(input, error)
  …
  optimMethod.optimize(model.weight,
model.gradient)
}
```

# BASELINE: PARAMETER SYNCHRONIZATION IN SPARK MLLIB



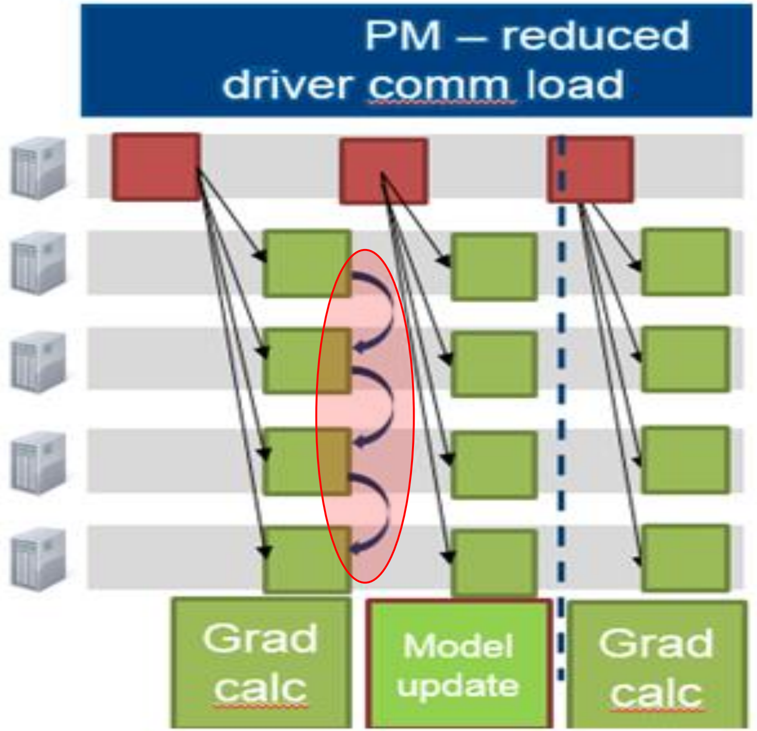

❷ Each task computes gradients

➊ Broadcast model parameters (weights) to each worker

❸ Each task sends gradients for (tree) aggregation, and

❹ then driver updates the weight

# SYNCHRONIZATION VIA PARAMETER MANAGER IN BIGDL



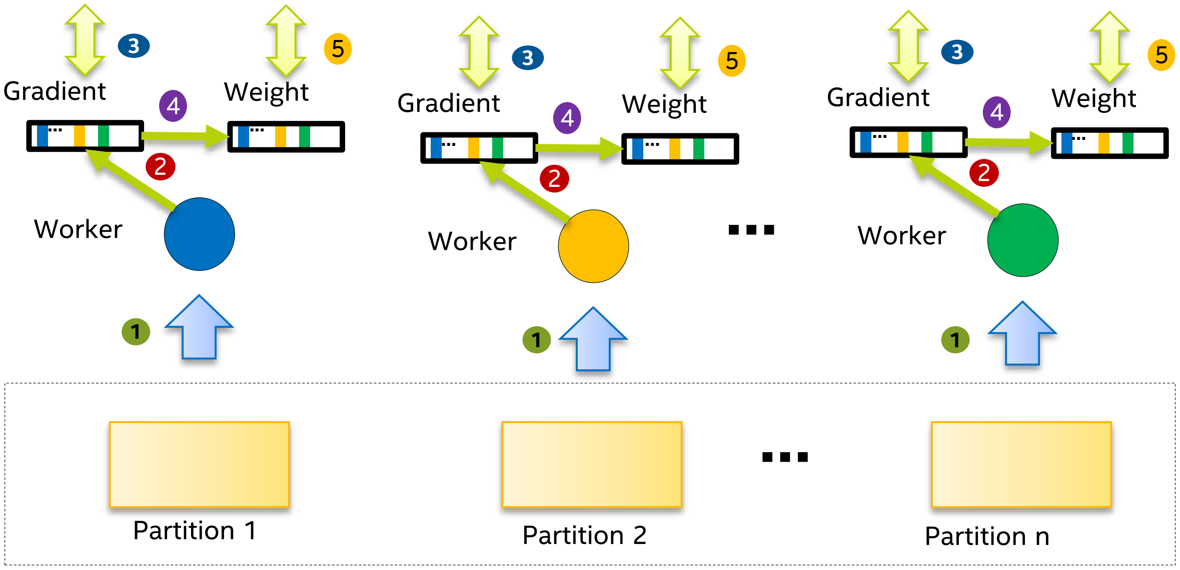

Distributed Parameter Manager aggregates gradients and updates model

**Master Node is not involved!**

**All-Reduce synchronization without hotspot and shuffle**

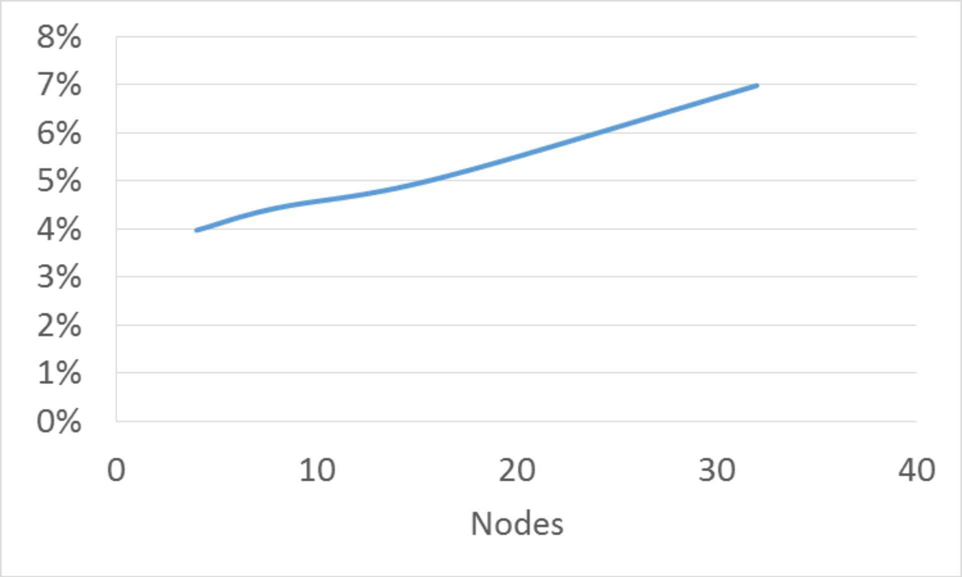
# SYNCHRONIZATION VIA PARAMETER MANAGER IN BIGDL

**PS (Parameter Server) Architecture in BigDL**
**on top of Spark Block Manager**




**Peer-2-Peer All-Reduce synchronization**

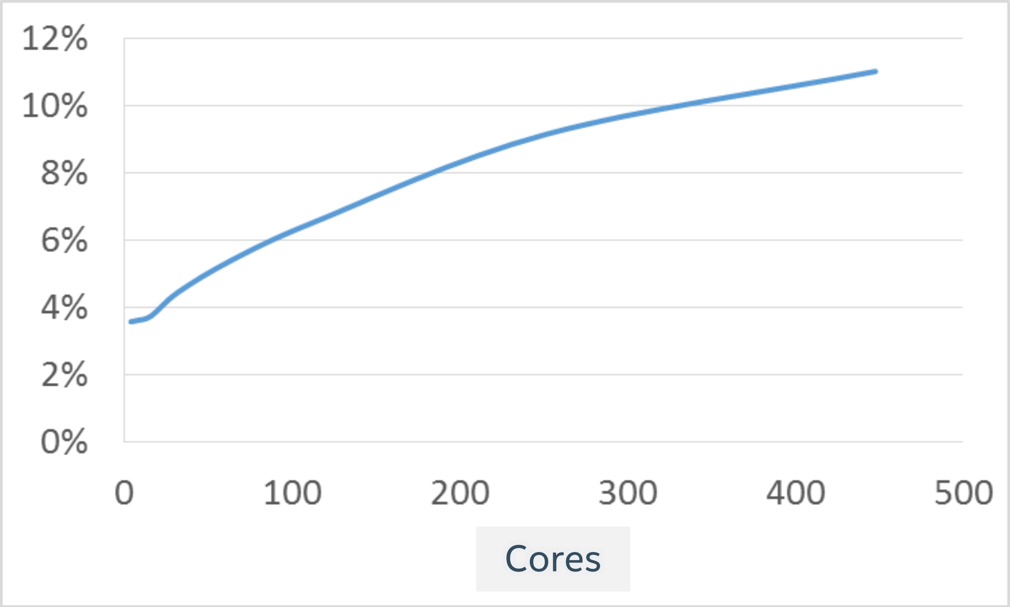
# EFFECTS OF PARAMETER MANAGER IMPLEMENTATION IN BIGDL




Parameter synchronization time as a fraction of average compute time for Inception v1 training

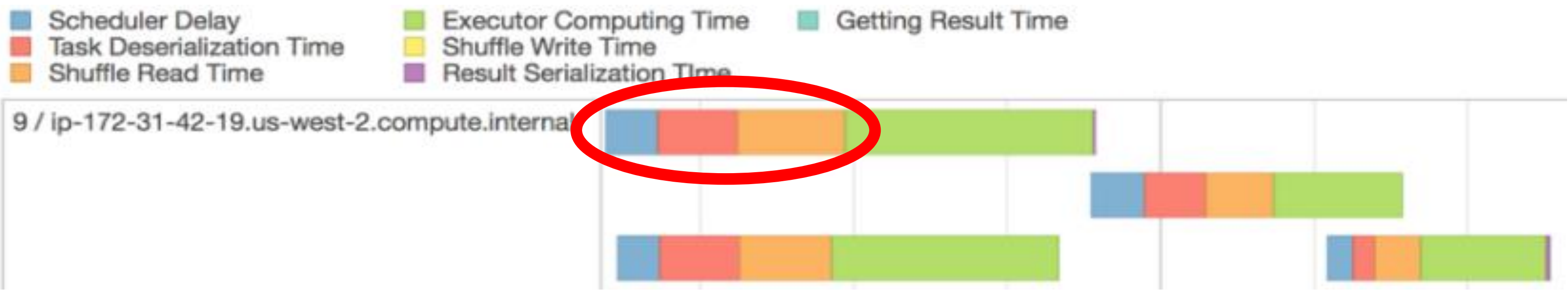
- Linear scaling
- 2x node increase – **only 1%** increase in parameter sync time

# TASK SCHEDULING OVERHEAD




Total Spark overhead (task scheduling, task serdes, task fetch) as a fraction of average compute time for Inception-v1 training

# FOCUS: CUTTING SPARK SCHEDULING AND COMMS OVERHEAD



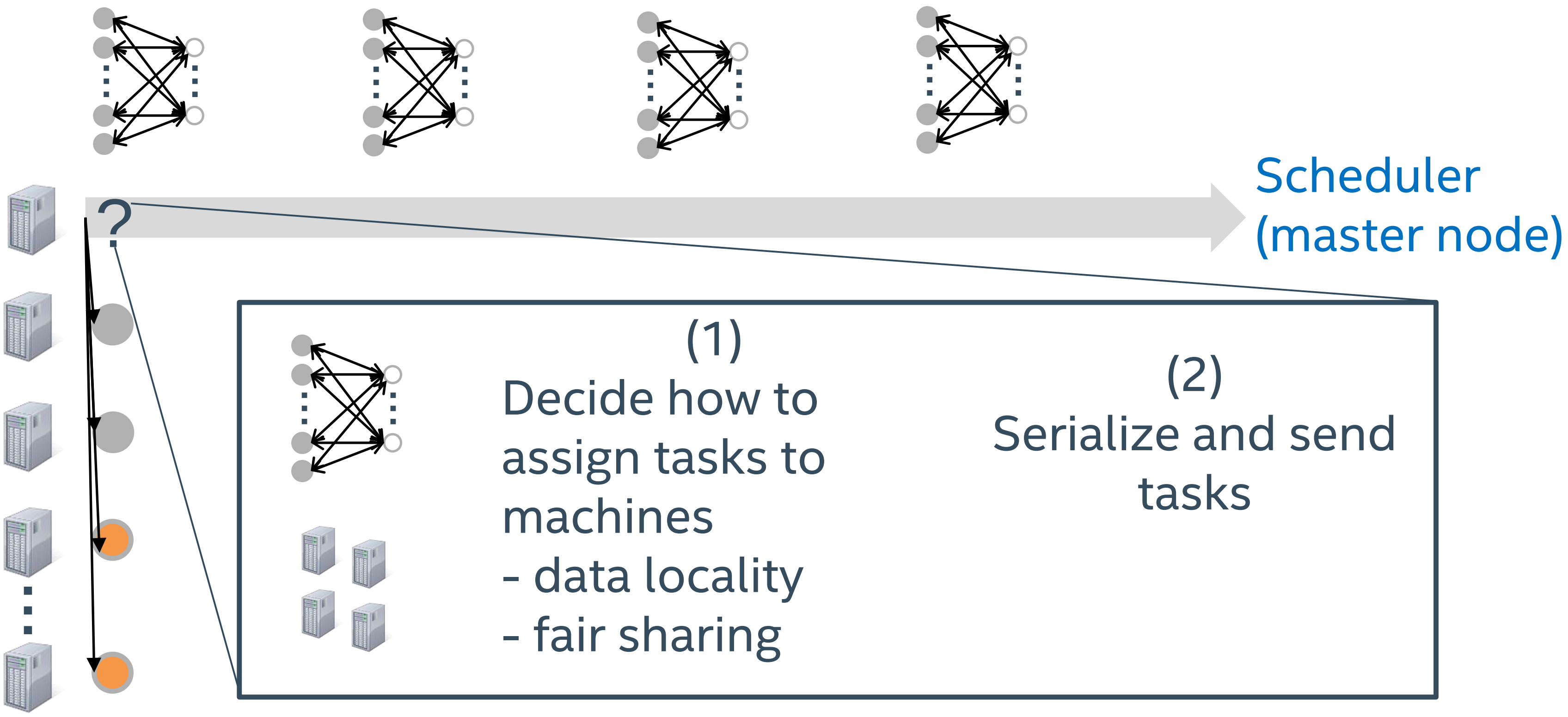

- Optimizing parameter synchronization and aggregation (PM)

- **Optimizing task scheduling (Drizzle)**

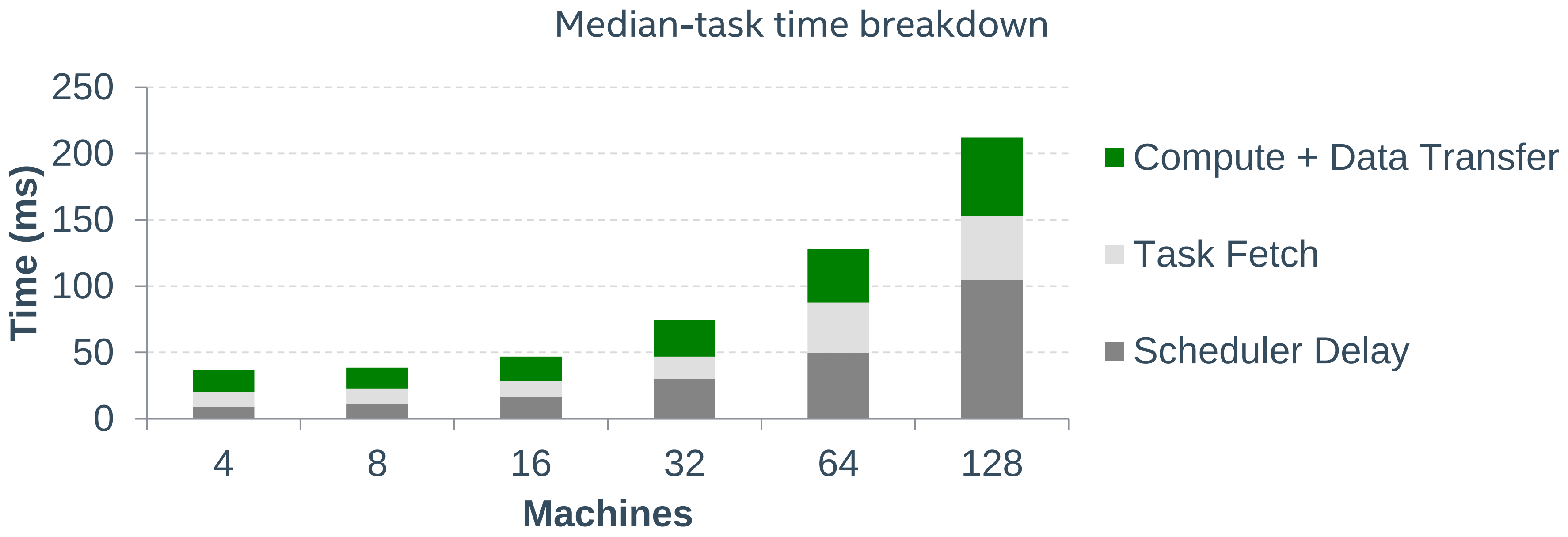
DL tasks are uniquely suited for Spark performance optimization:

- Heavy master node workload during model update
- **Repetitive in nature (reusable task scheduling decisions)**
- **Static data partitioning and cluster configuration**

# INSIDE THE SCHEDULER



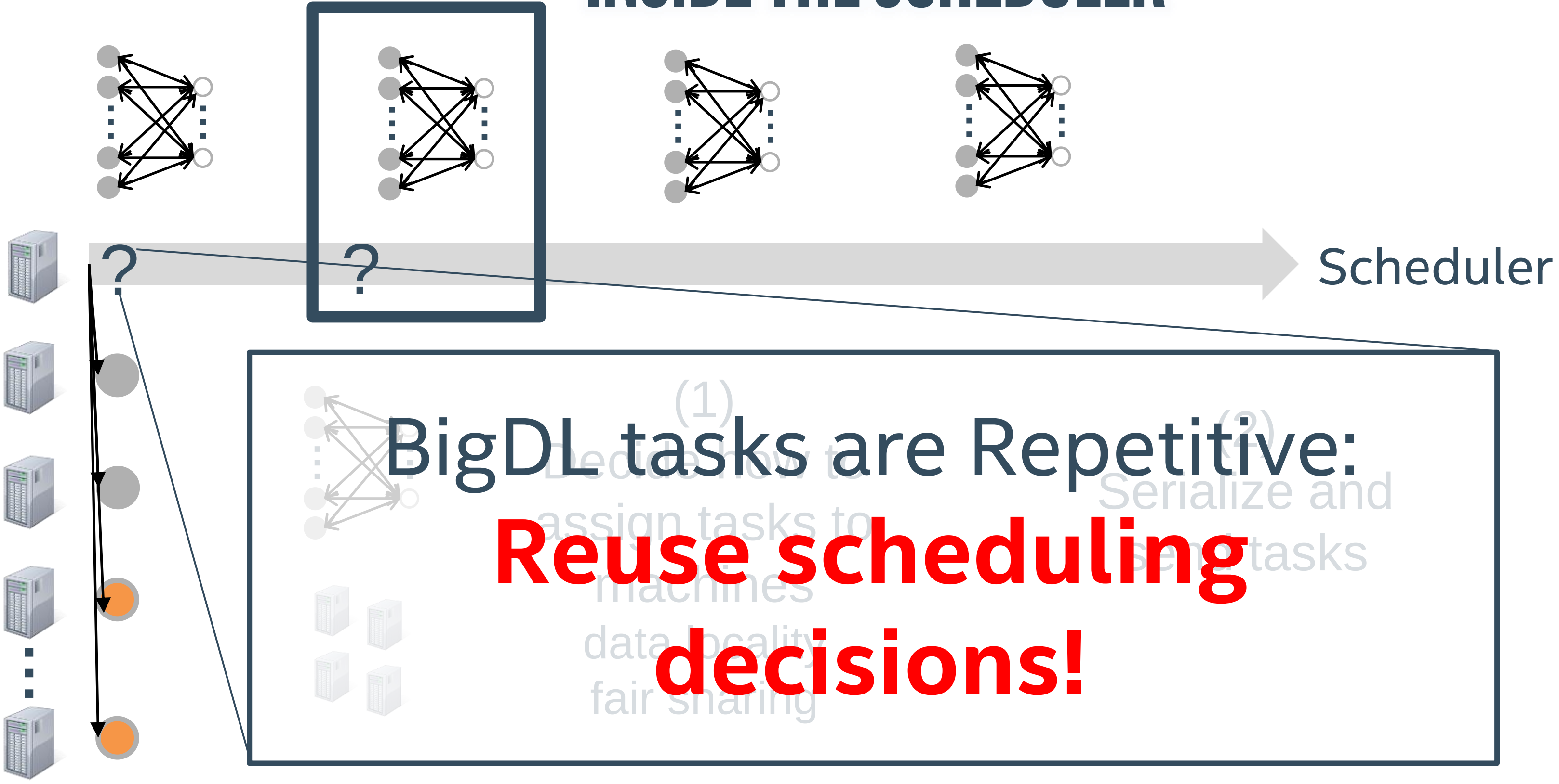
# SCHEDULING OVERHEADS – SCALABILITY PROBLEM




Cluster: 4 core, r3.xlarge machines

Workload: Sum of 10k numbers per-core

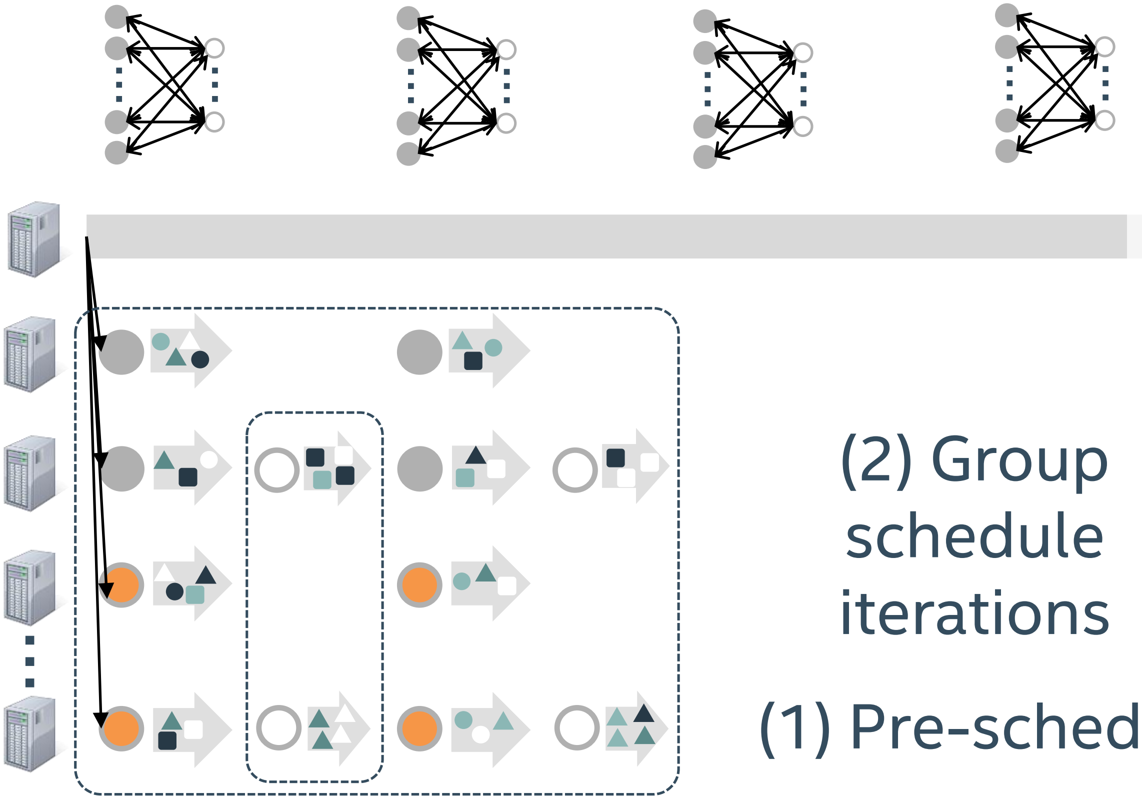
# INSIDE THE SCHEDULER




BigDL tasks are Repetitive:

**Reuse scheduling decisions!**

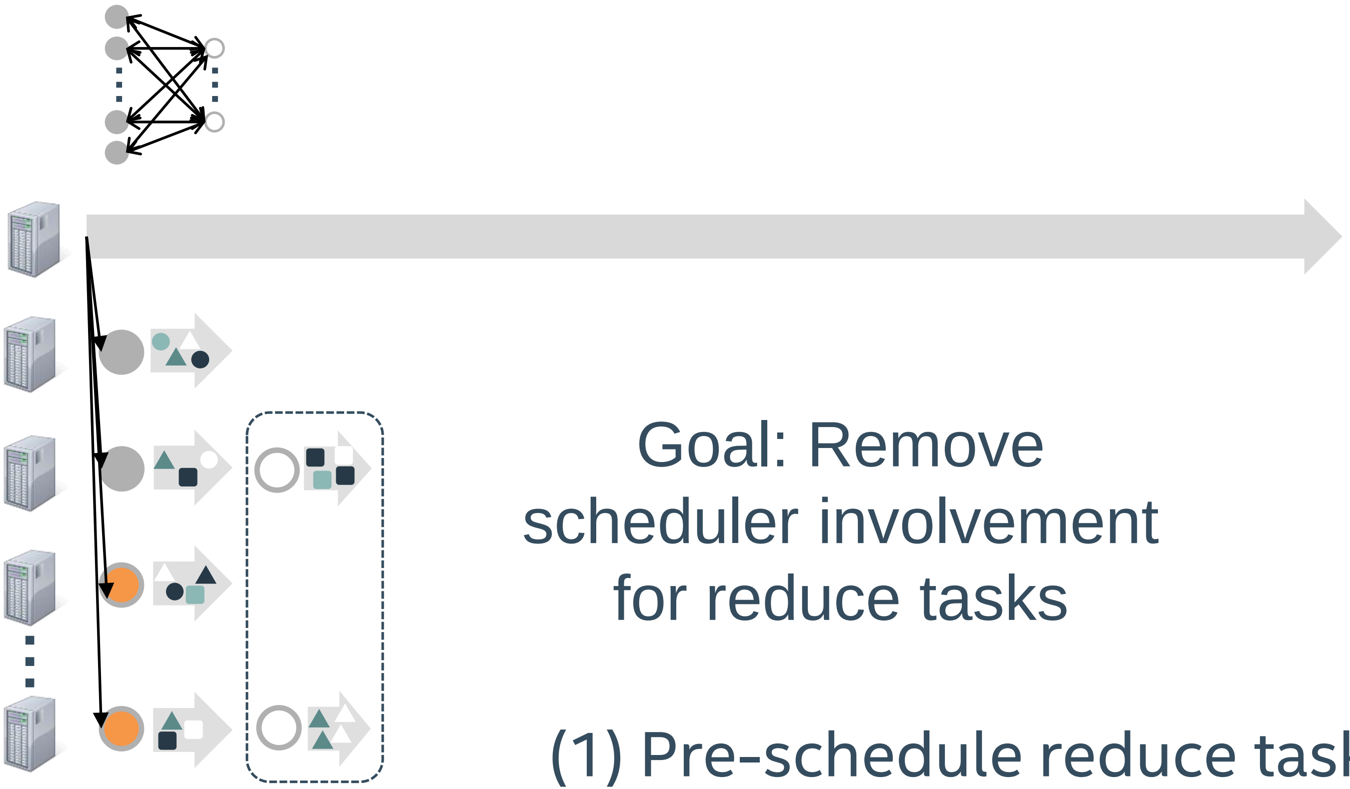
# DRIZZLE



Goal:
remove frequent scheduler interaction

(2) Group schedule iterations

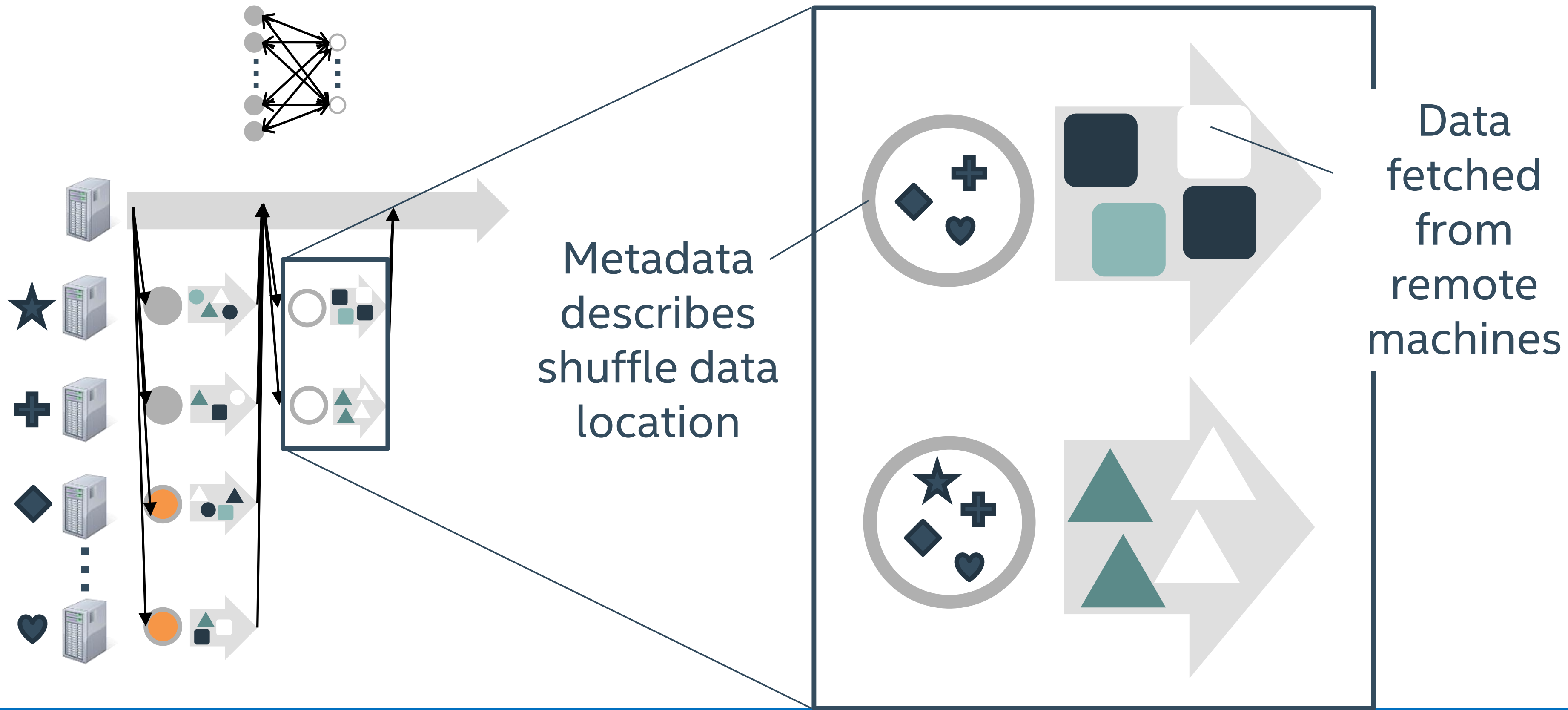
(1) Pre-schedule reduce tasks



Goal: Remove scheduler involvement for reduce tasks

(1) Pre-schedule reduce tasks

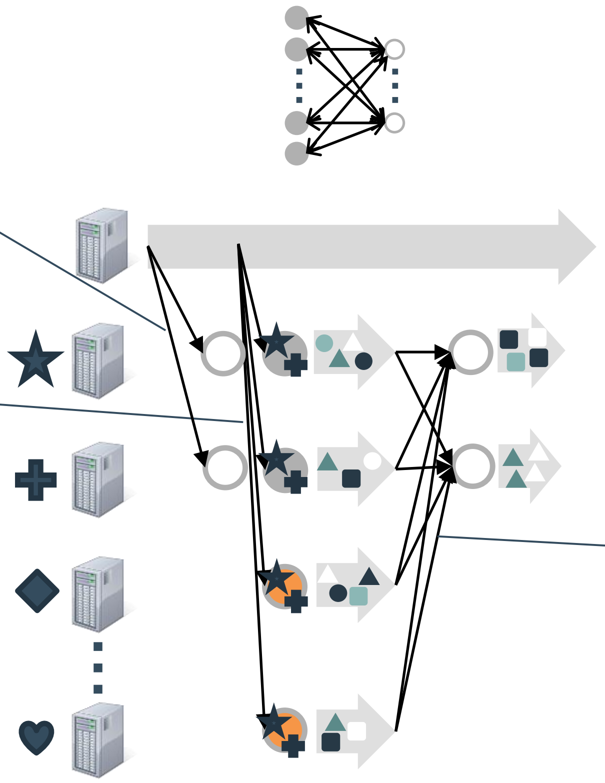
# COORDINATING SHUFFLES: EXISTING SYSTEMS



# COORDINATING SHUFFLES: PRE-SCHEDULING

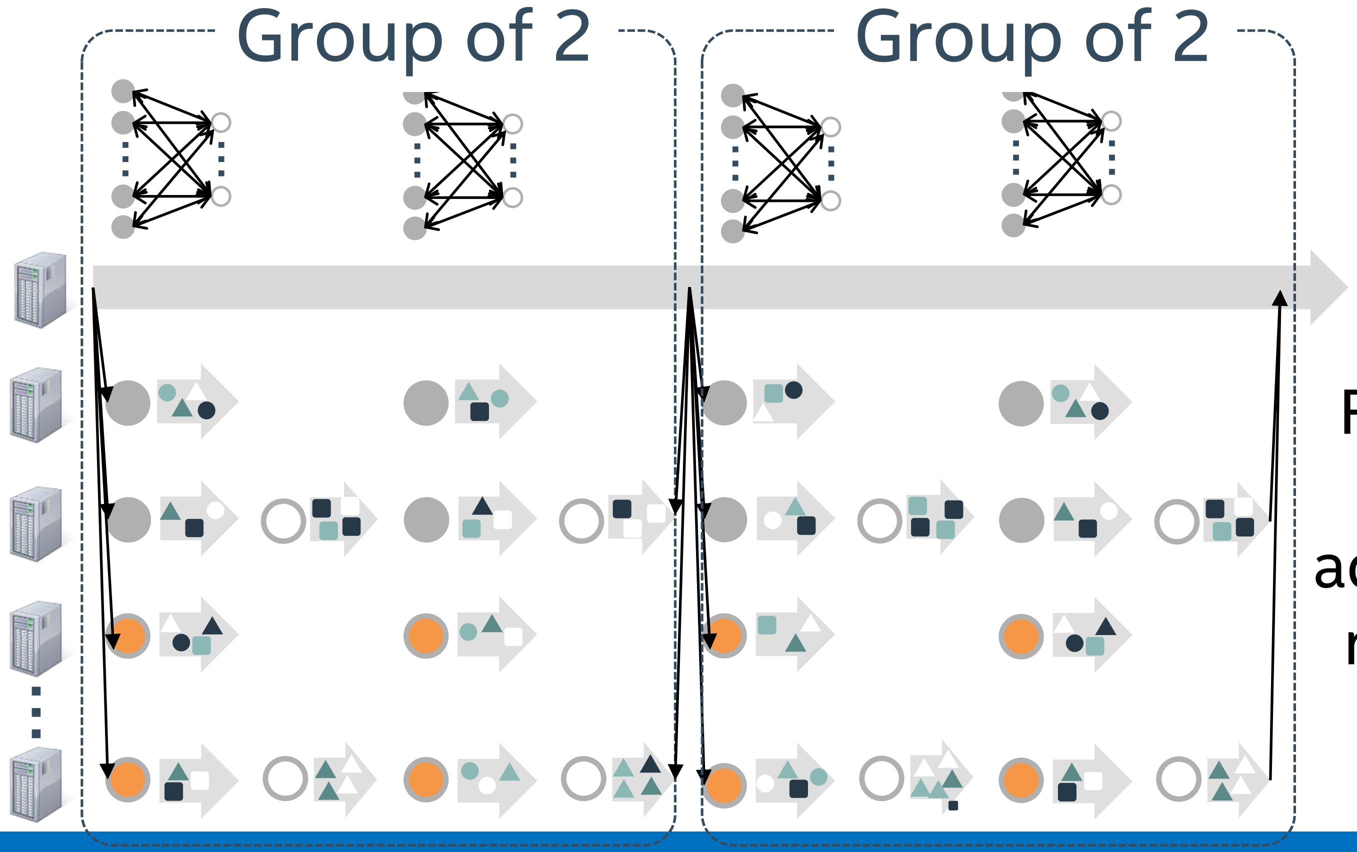
(1) Pre-schedule reducers

(2) Mappers get metadata



(3) Mappers trigger reducers

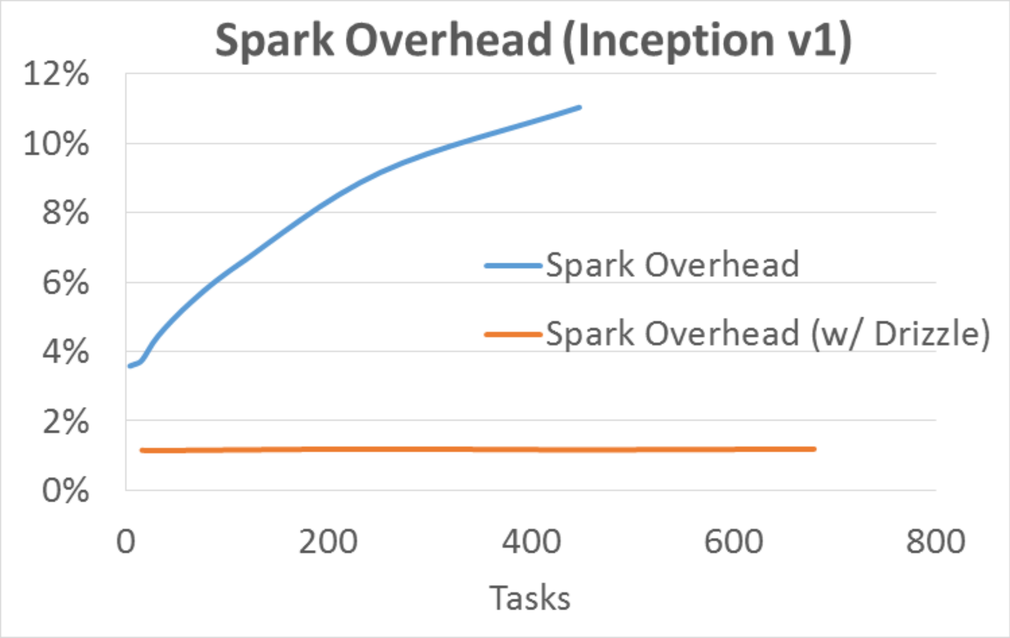
# GROUP SCHEDULING




Schedule **group** of iterations at once

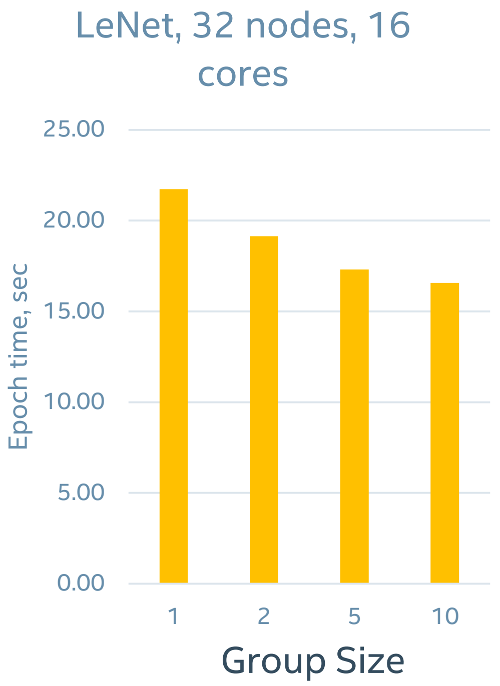
Fault tolerance, scheduling, adding/removing nodes at group boundaries

# REDUCING SCHEDULING OVERHEADS WITH DRIZZLE

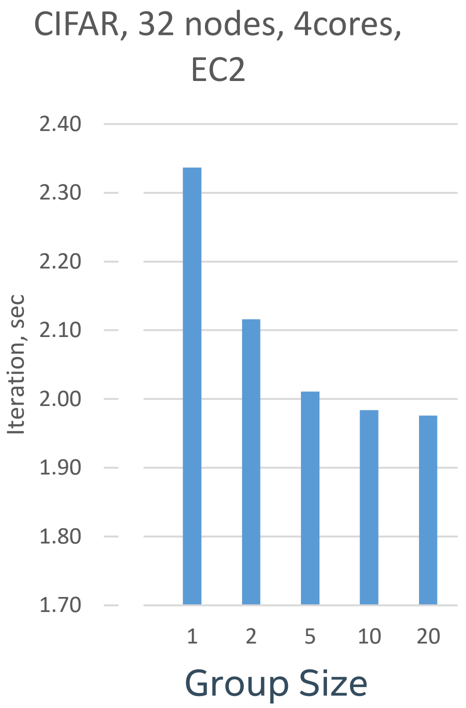


# DRIZZLE – BIGDL PERFORMANCE IMPROVEMENT

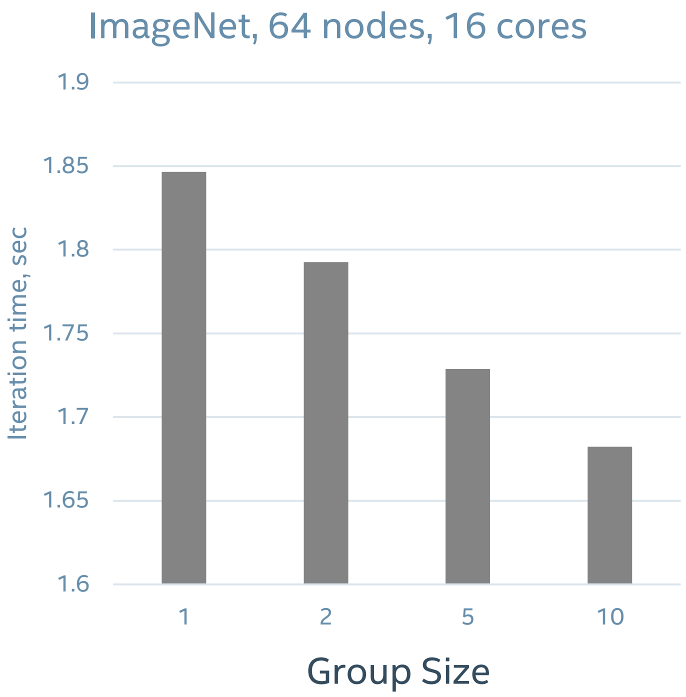
## YOUR MILEAGE WILL VARY…




15% improvement




24% improvement




10% improvement

# CONCLUSION

- Deep Learning Spark jobs are somewhat unique
  - Heavy master node load for large model parameter update
  - Relatively short execution tasks (for fast model conversion)
  - Scheduling/Comms sometimes takes ~50% of total task execution.

- Deep Learning tasks are uniquely suited for optimization

  * Distributed Parameter Manager to offload Master compute.

  * Drizzle takes advantage of repetitive nature of the tasks and static data partitioning.

* Need Spark committers community involvement

# FURTHER READING ☺

https://github.com/intel-analytics/BigDL/tree/new_parametermanager_drizzle

https://github.com/amplab/drizzle-spark

http://shivaram.org/publications/drizzle-sosp17.pdf

# LEGAL NOTICES & DISCLAIMERS