



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

Shivaram Venkataraman (Microsoft Research, UC Berkeley)

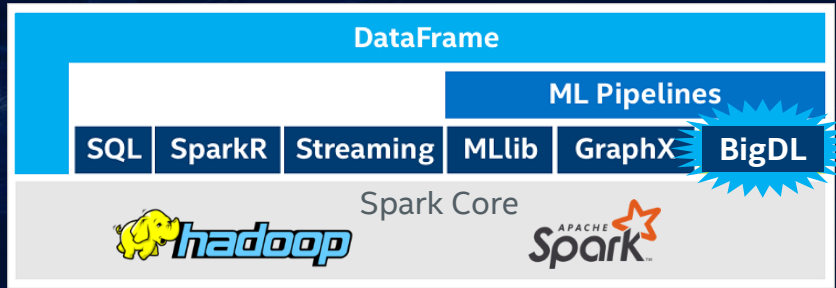
Ding Ding (Intel)

Sergey Ermolin (Intel)

June 2018



HIGH PERFORMANCE DEEP LEARNING FOR APACHE SPARK* ON CPU INFRASTRUCTURE



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**

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

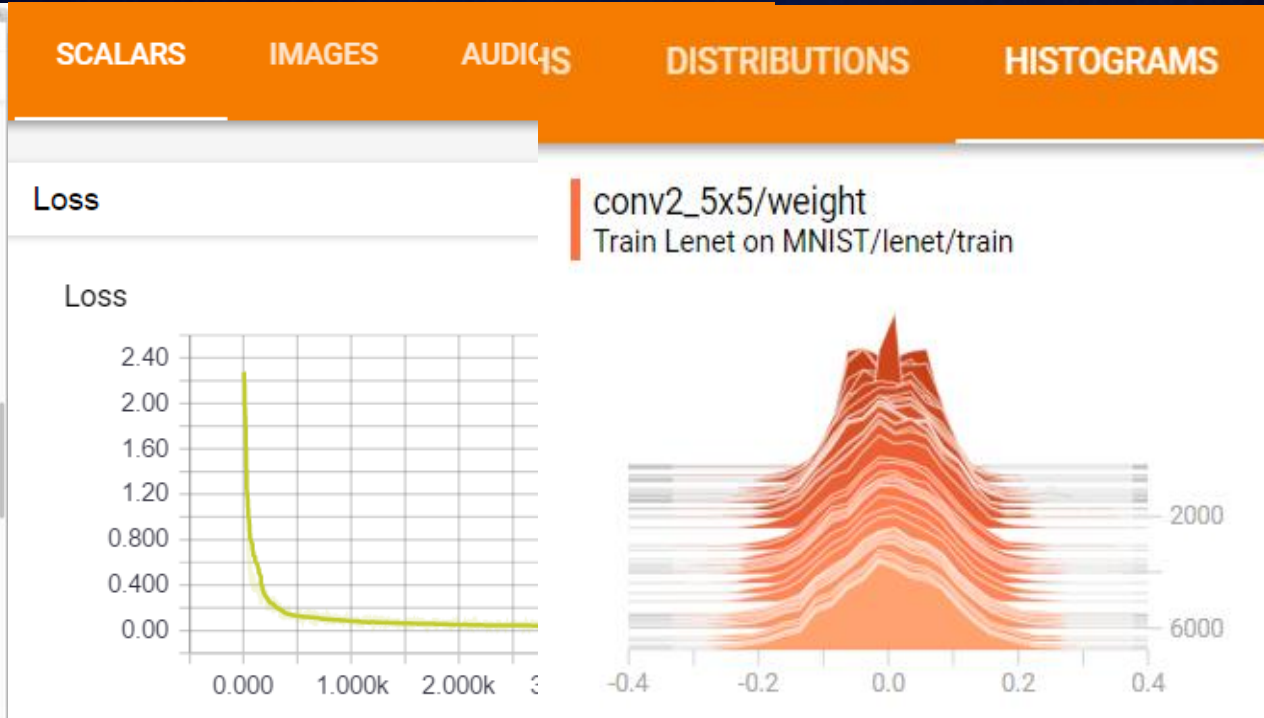
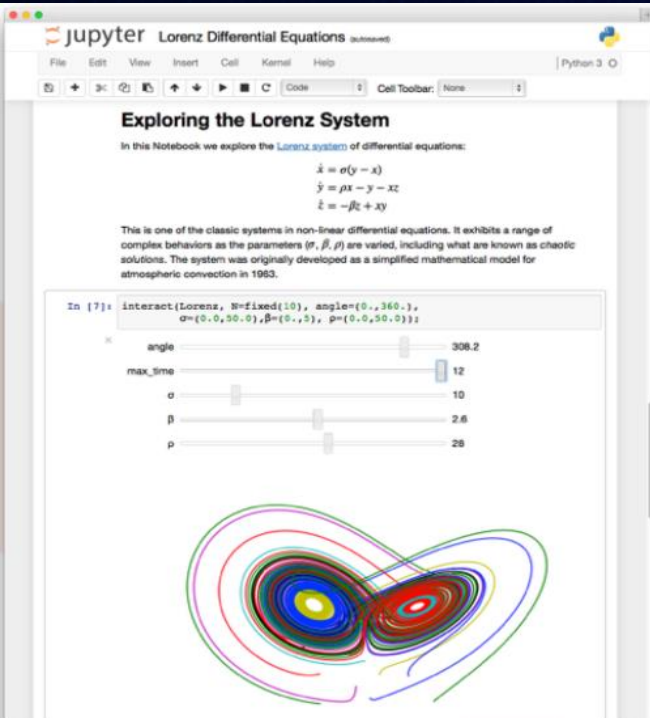
Powered by Intel® MKL and multi-threaded programming

bigdl-project.github.io

software.intel.com/bigdl



Jupyter, Zeppelin notebooks and TensorBoard support



BUILDING & DEPLOYING WITH BIGDL

PLATFORMS



CLOUD SERVICE PROVIDERS



SOLUTIONS



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

<https://bigdl-project.github.io>

And Many More...

ANALYTICS ZOO

Analytics + AI Pipelines for Spark and BigDL

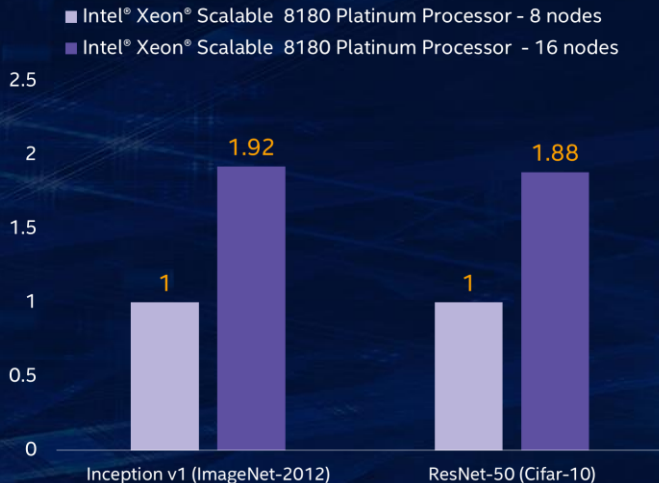
“Out-of-the-box” ready for use

- 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.

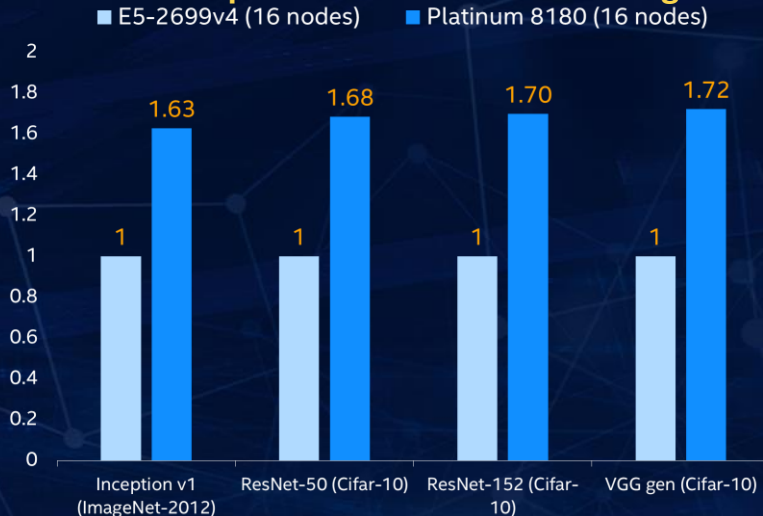


DEEP LEARNING WITH BIGDL/SPARK

Node Scaling with BigDL



Generational performance increase with BigDL

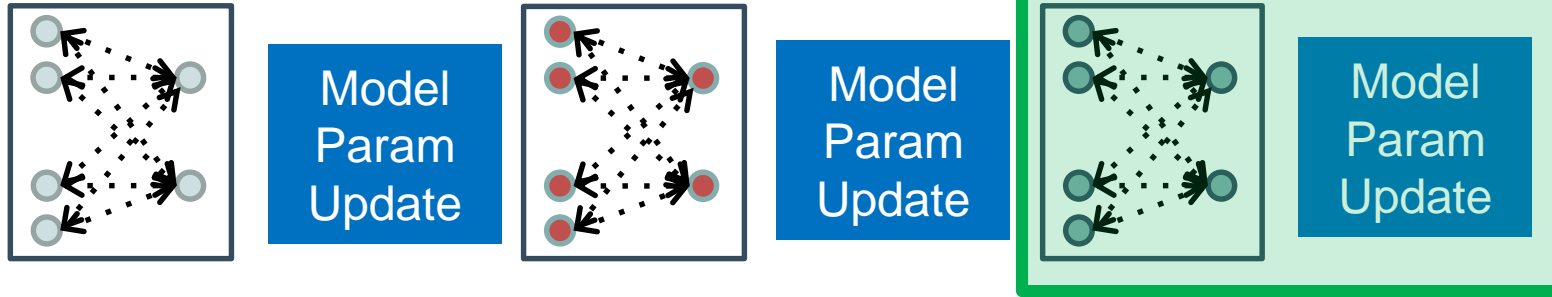


**GET EXCELLENT MULTI-NODE SCALING AND GENERATIONAL PERFORMANCE
WITH YOUR EXISTING HARDWARE**

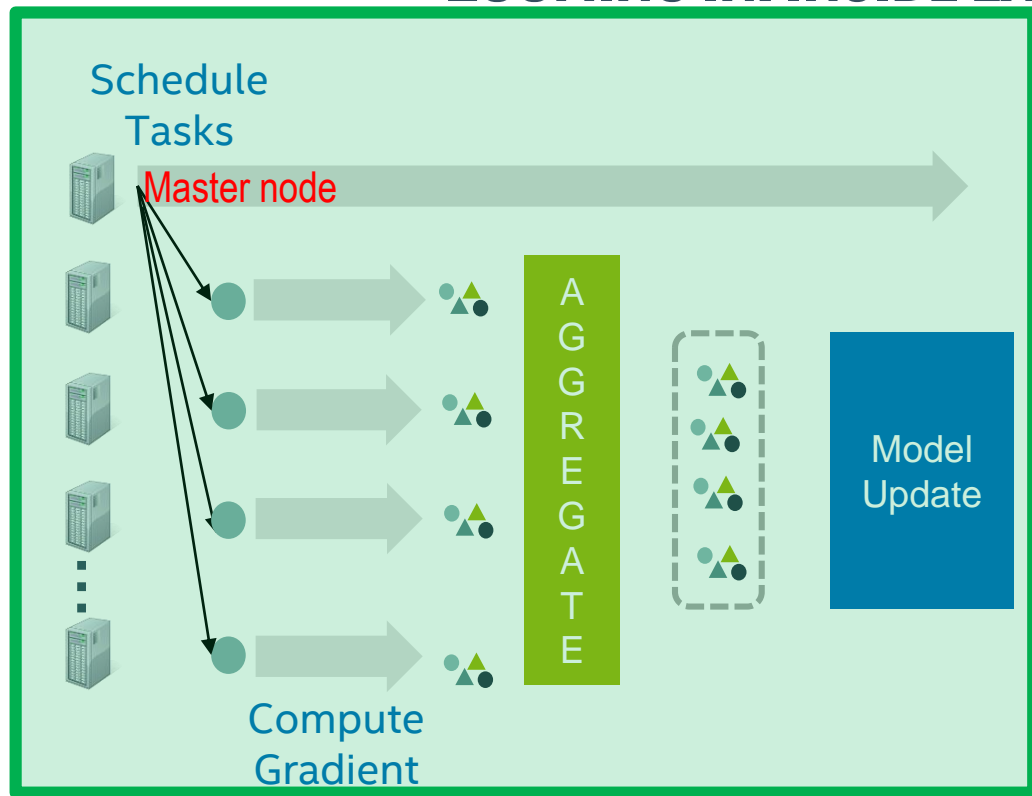
Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Benchmark results were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown". Implementation of these updates may make these results inapplicable to current and future workloads and performance tests. Some workloads may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark, may not be representative of actual workloads. Performance may vary due to many factors and functions. Any of those factors may cause the results to vary. You should consult the information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit <http://www.intel.com/performance>.
bigdl-project.github.io software.intel.com/bigdl

DEEP LEARNING TRAINING

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



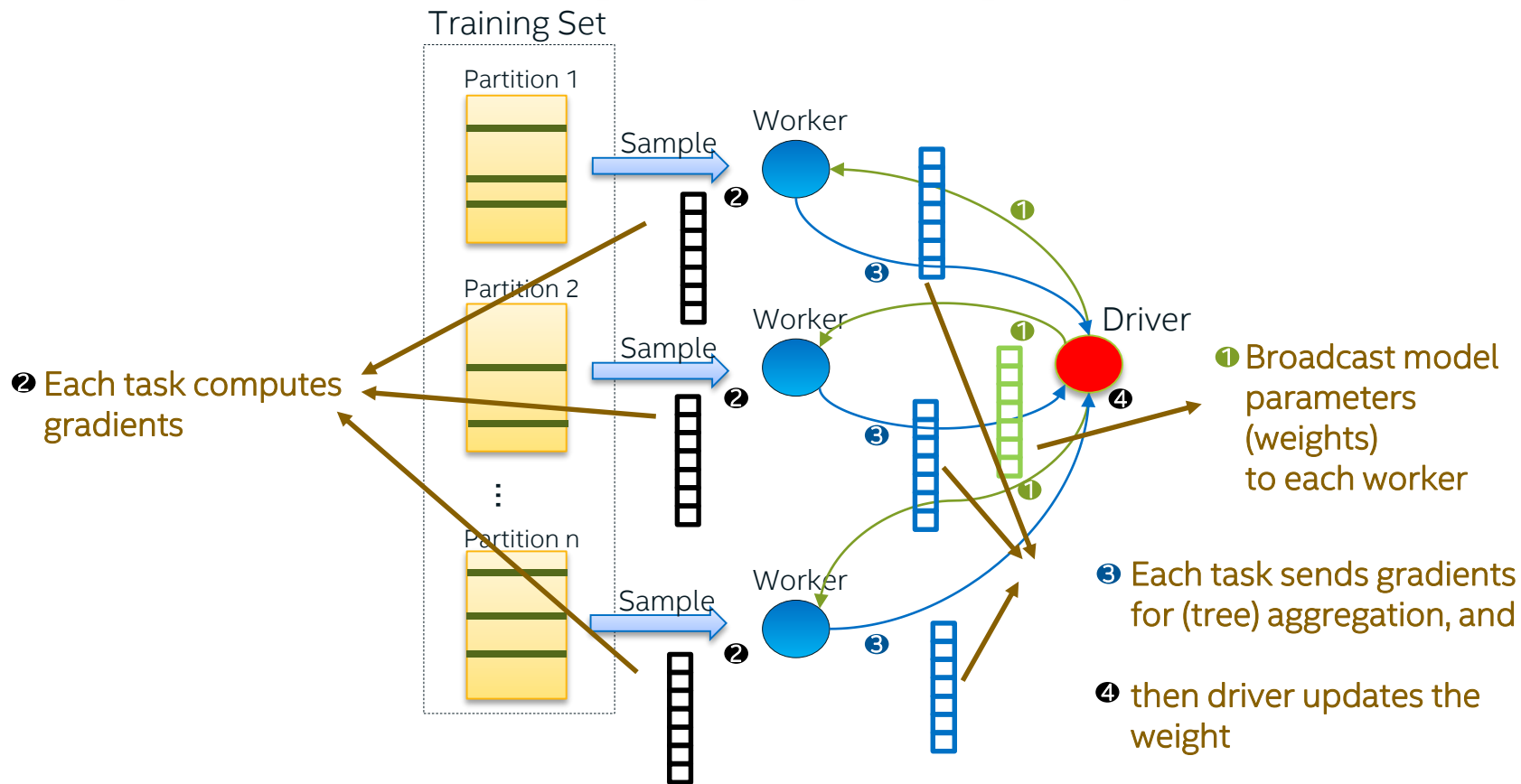
ZOOMING IN: INSIDE EACH ITERATION



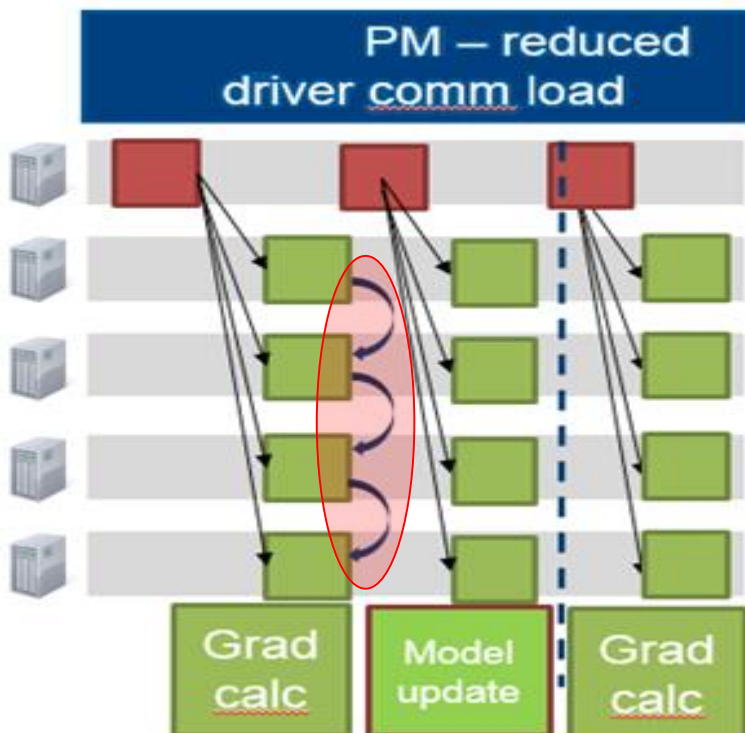
Training

```
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 ML LIB



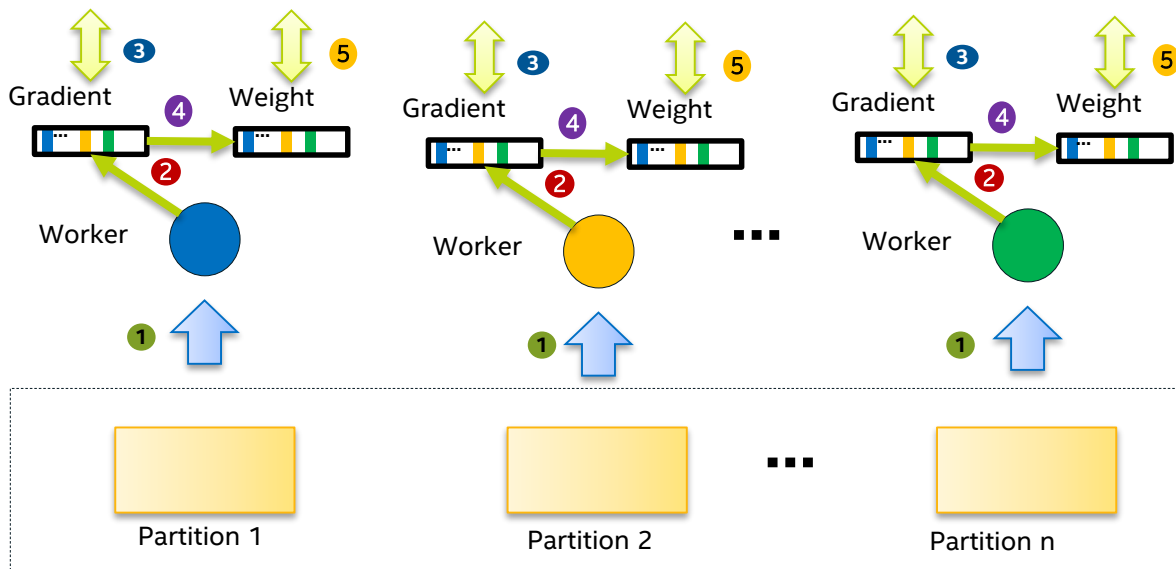
SYNCHRONIZATION VIA **PARAMETER MANAGER** IN BIGDL



All-Reduce synchronization without hotspot and shuffle

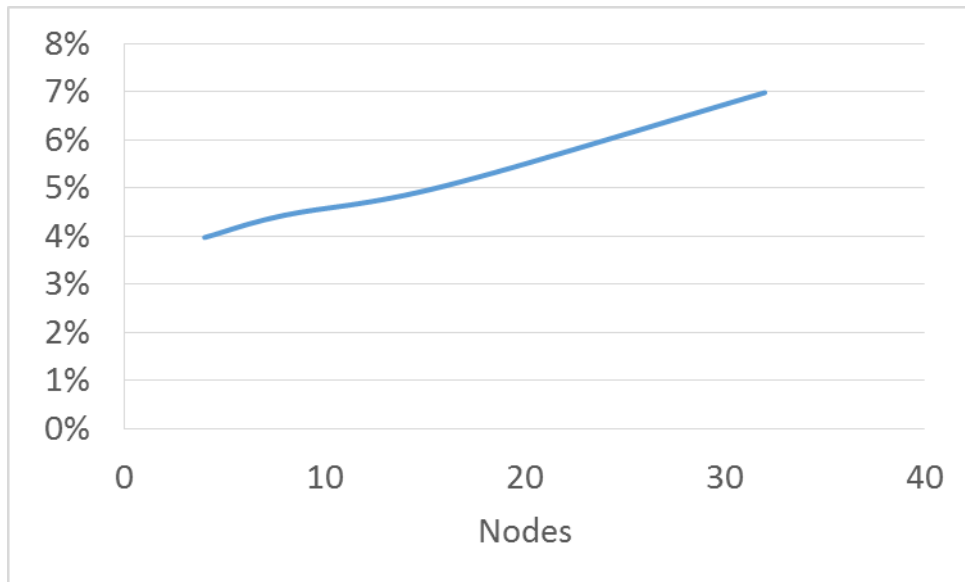
SYNCHRONIZATION VIA **PARAMETER MANAGER** IN BIGDL

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



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

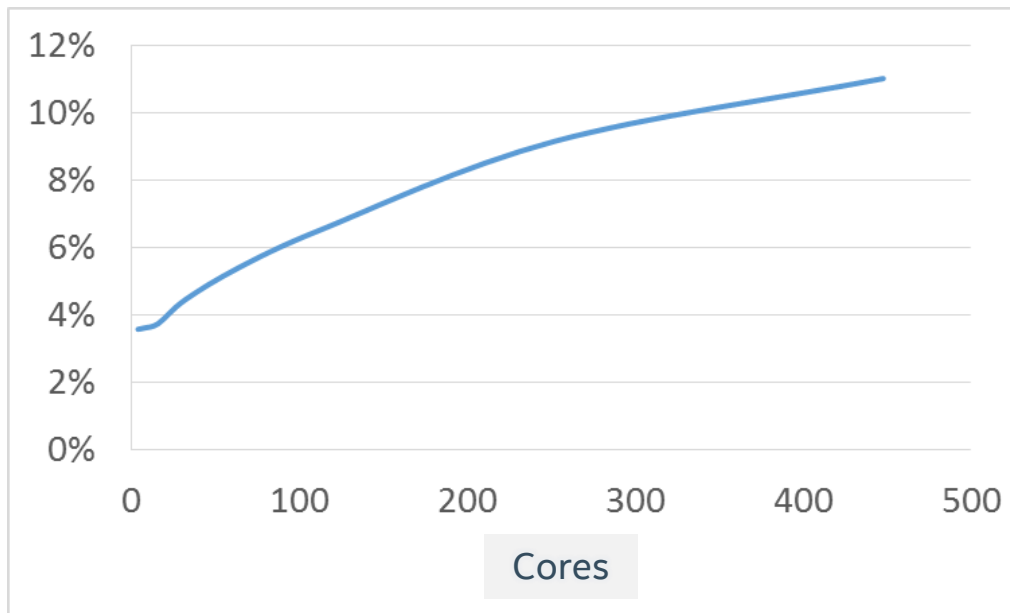
EFFECTS OF PARAMETER MANAGER IMPLEMENTATION IN BIGDL



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

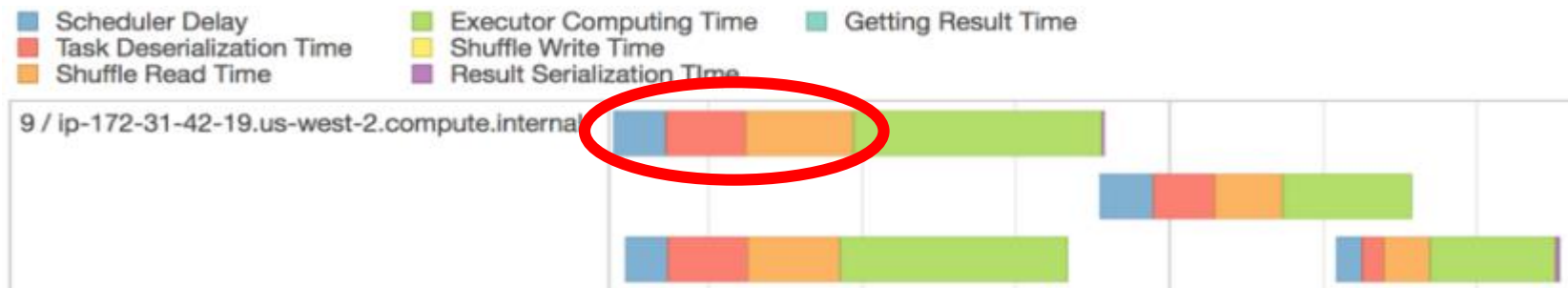
- Linear scaling
- 2x node increase – **only 1%** increase in parameter sync time

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

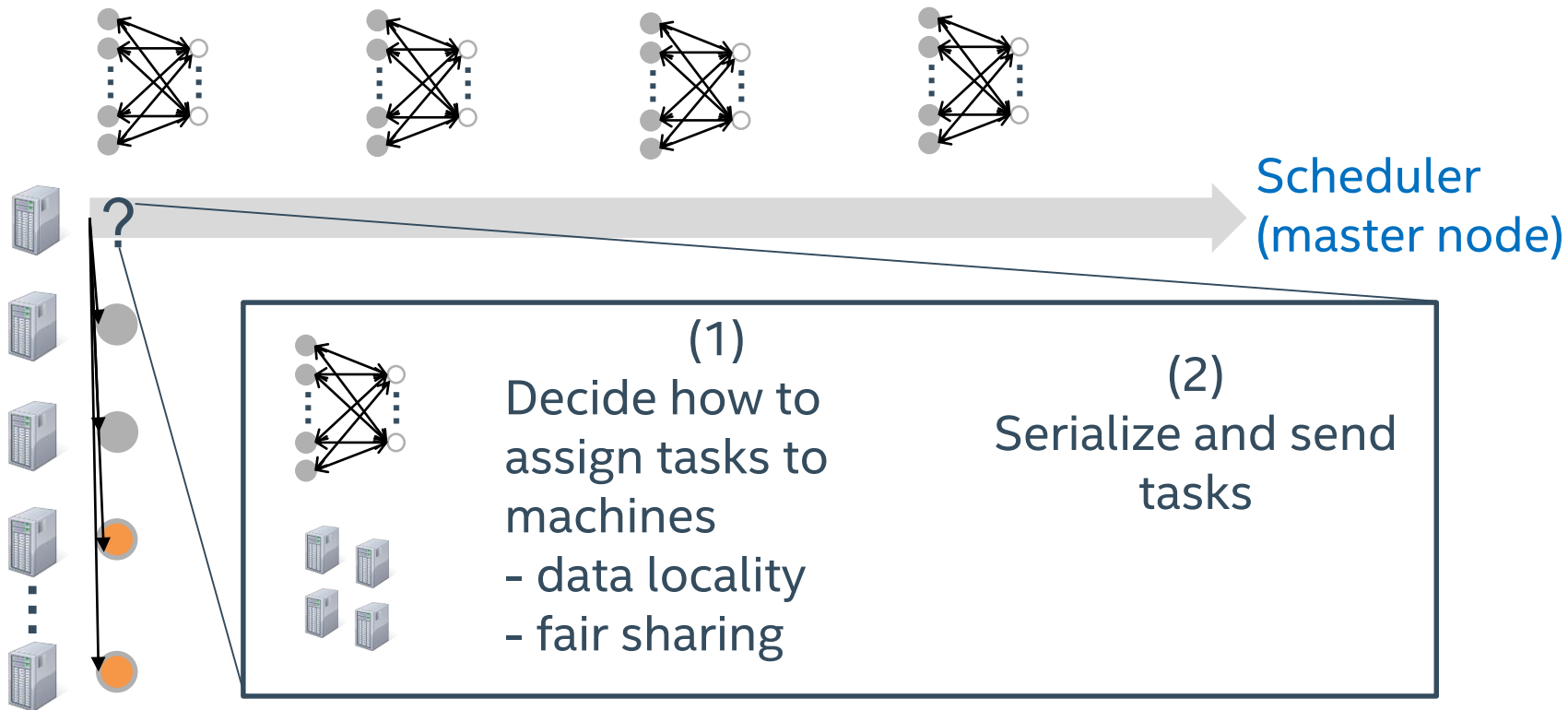


- Optimizing parameter synchronization and aggregation (PM)
- **Optimizing task scheduling (Drizzle)**

DL tasks are uniquely suited for Spark performance optimization:

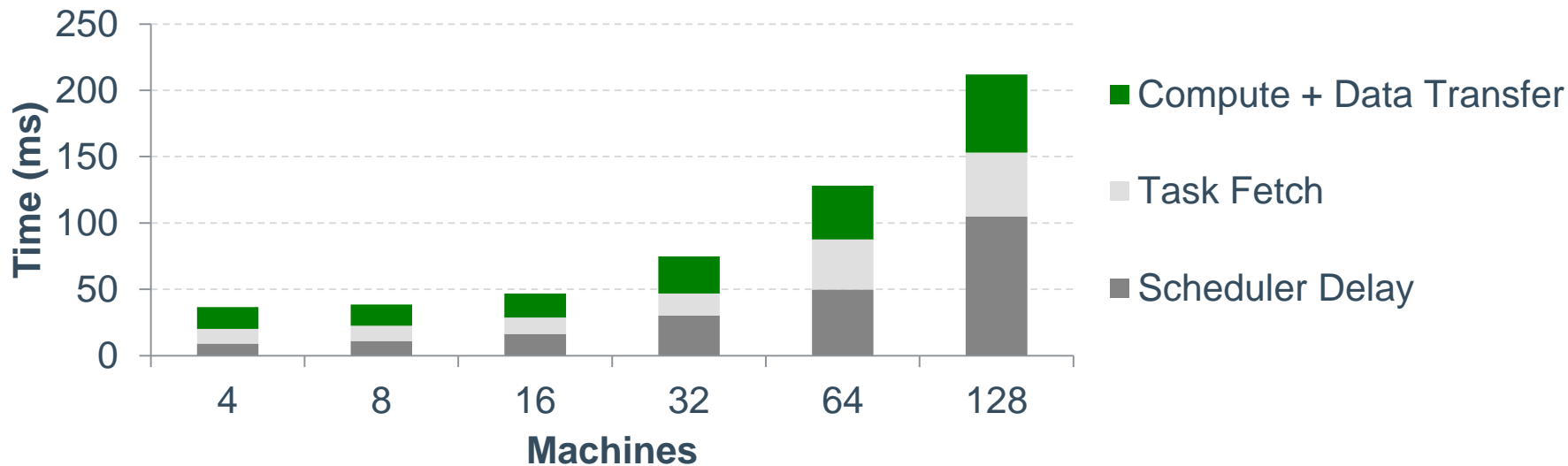
- Heavy master node workload during model update
- **Repetitive in nature (reusable task scheduling decisions)**
- **Static data partitioning and cluster configuration**

INSIDE THE SCHEDULER



SCHEDULING OVERHEADS – SCALABILITY PROBLEM

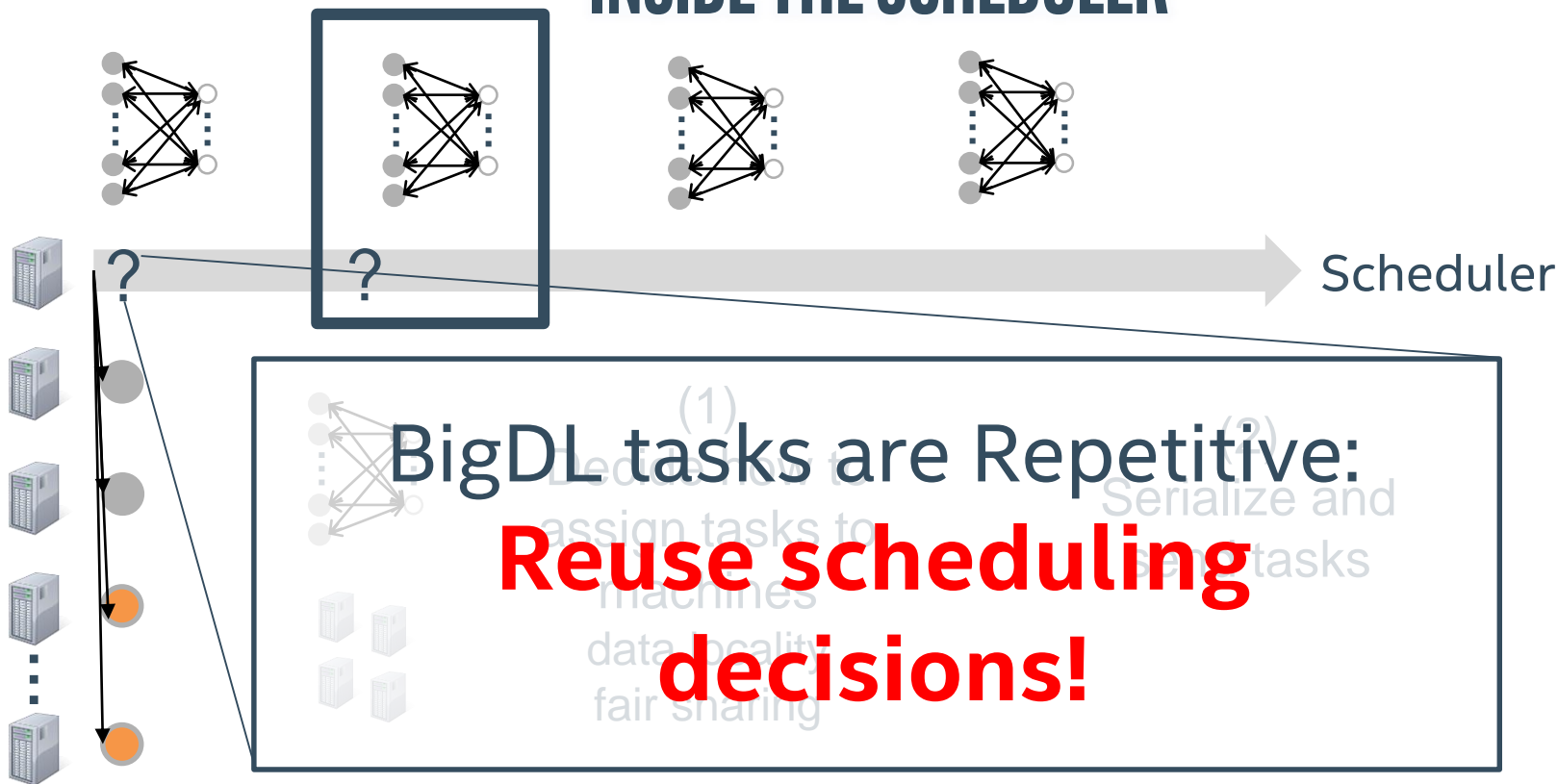
Median-task time breakdown



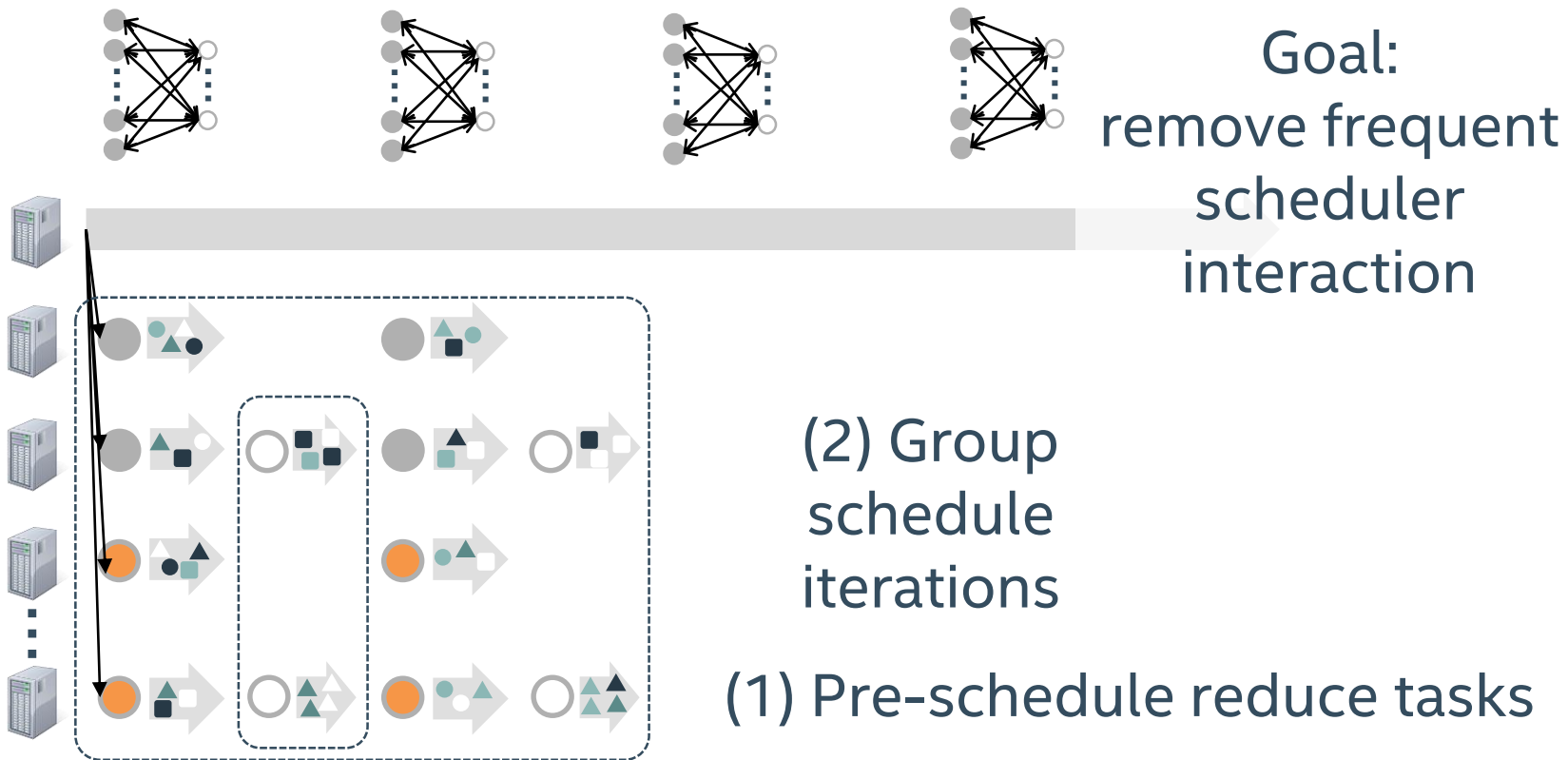
Cluster: 4 core, r3.xlarge machines

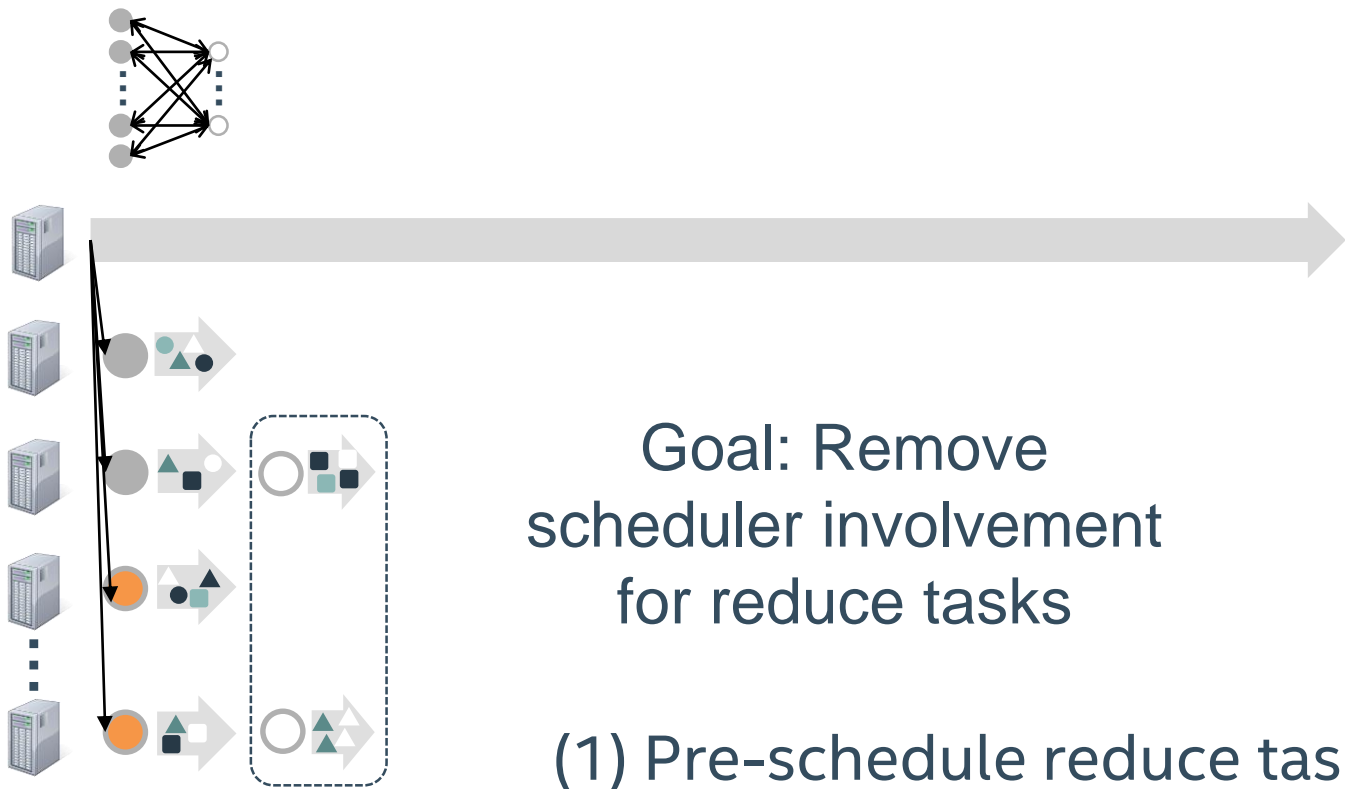
Workload: Sum of 10k numbers per-core

INSIDE THE SCHEDULER

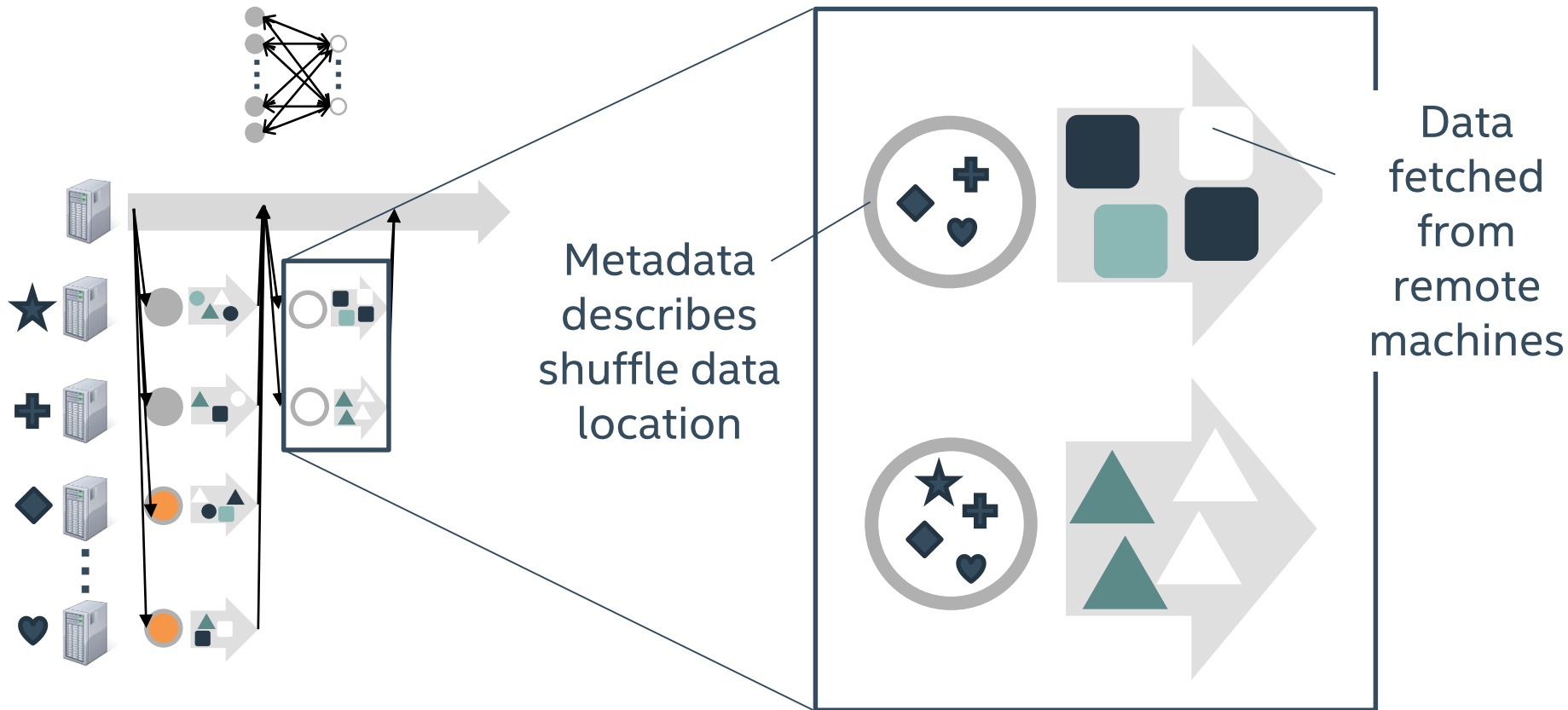


DRIZZLE

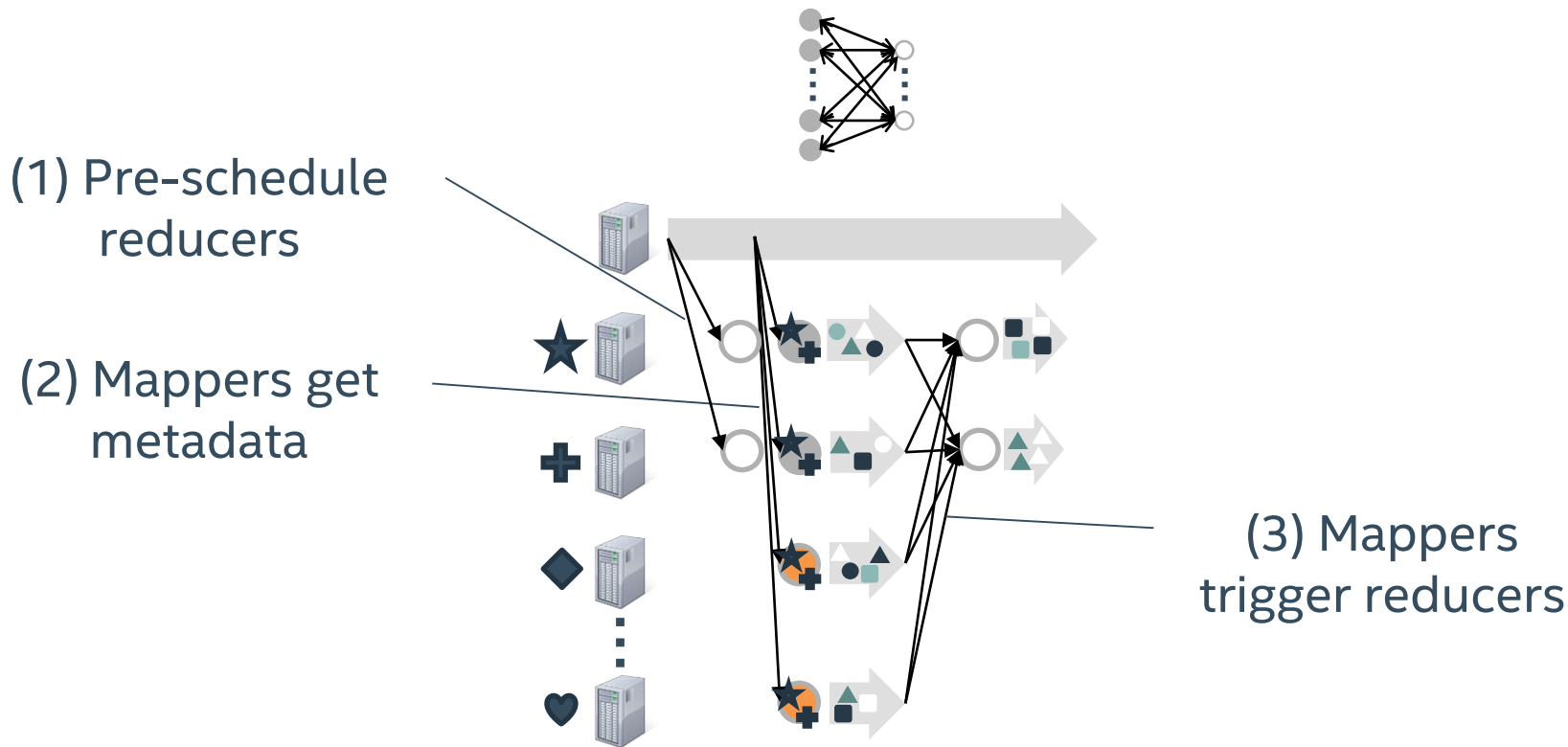




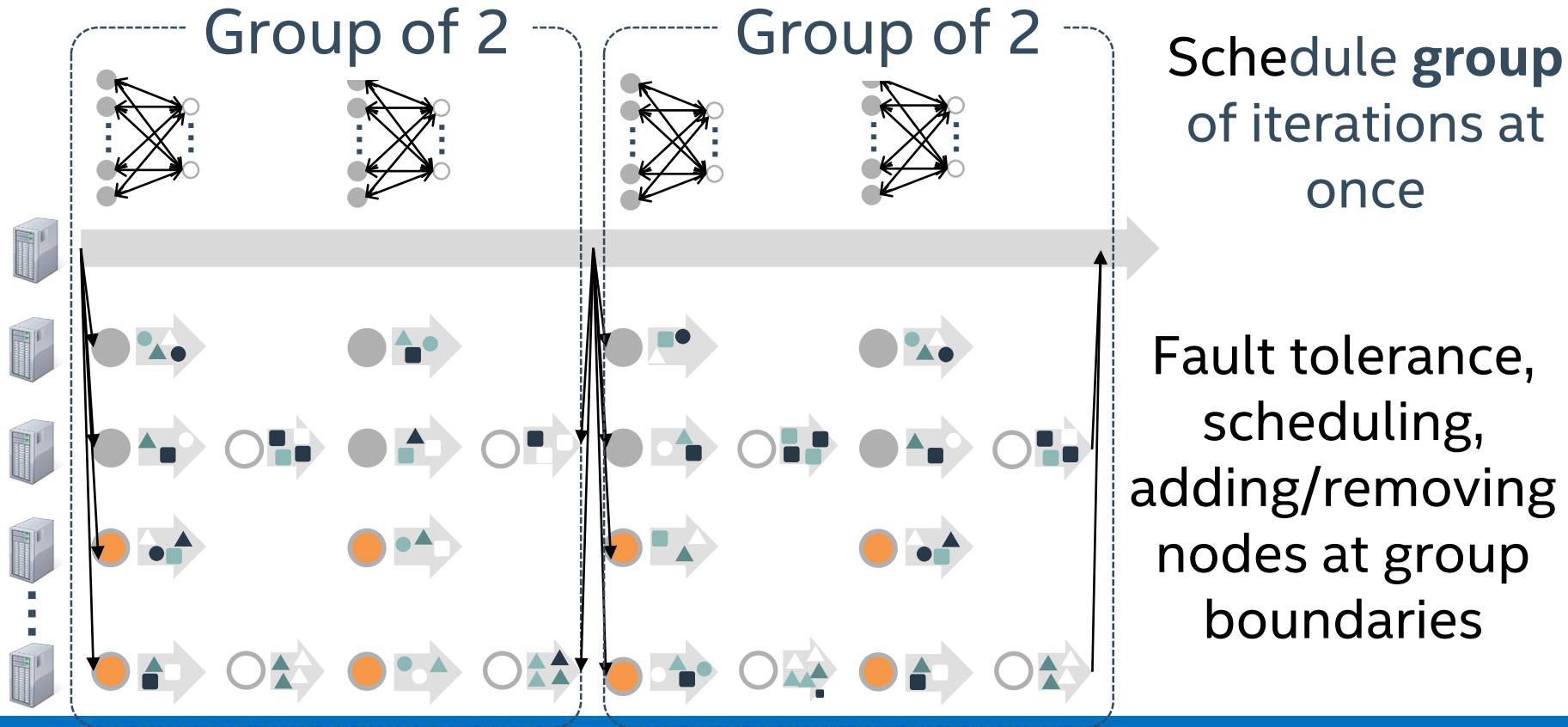
COORDINATING SHUFFLES: EXISTING SYSTEMS



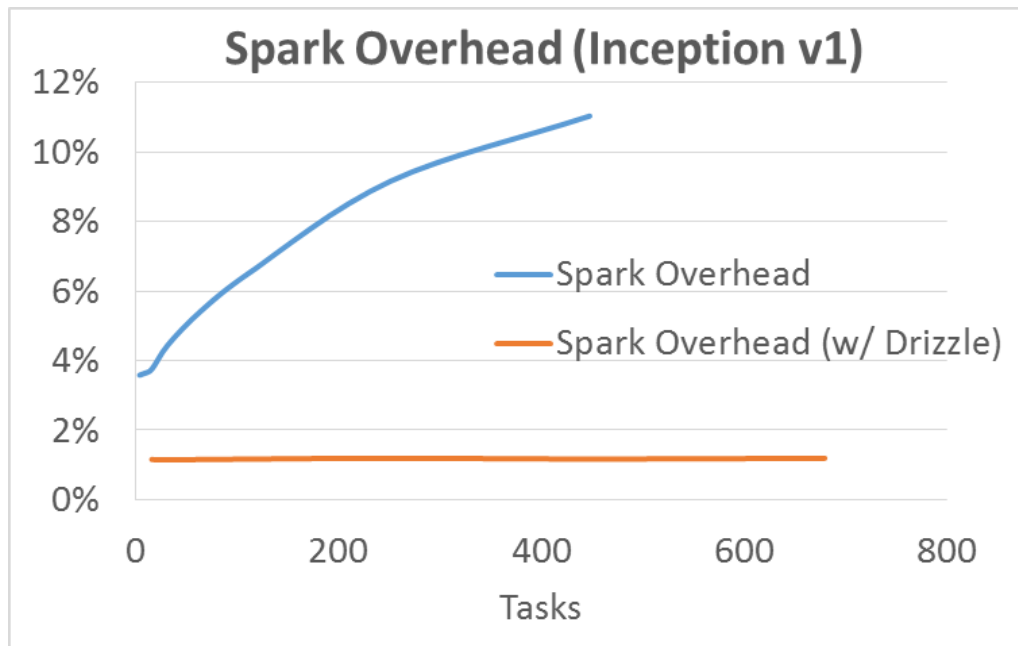
COORDINATING SHUFFLES: PRE-SCHEDULING



GROUP SCHEDULING



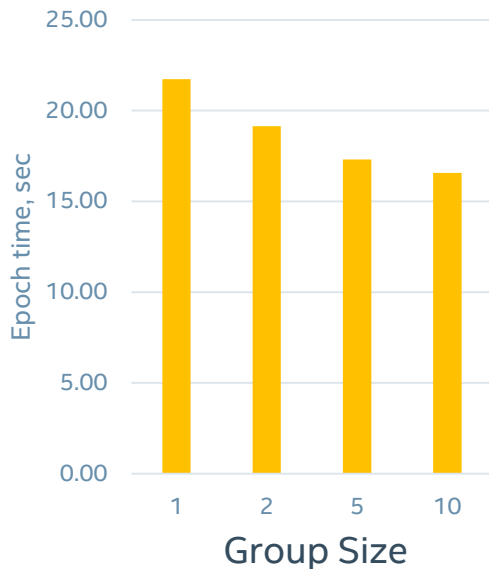
REDUCING SCHEDULING OVERHEADS WITH DRIZZLE



DRIZZLE - BIGDL PERFORMANCE IMPROVEMENT

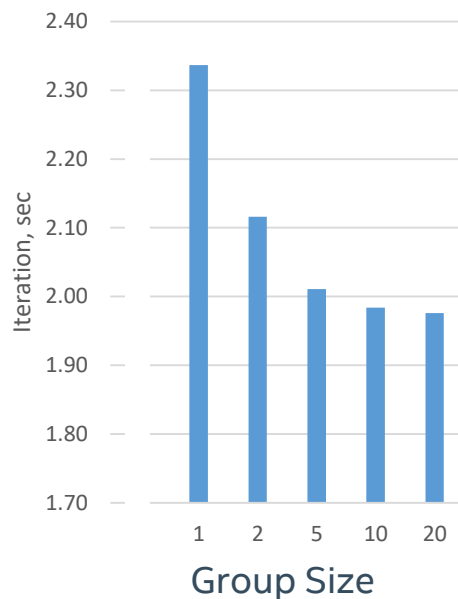
YOUR MILEAGE WILL VARY...

LeNet, 32 nodes, 16
cores



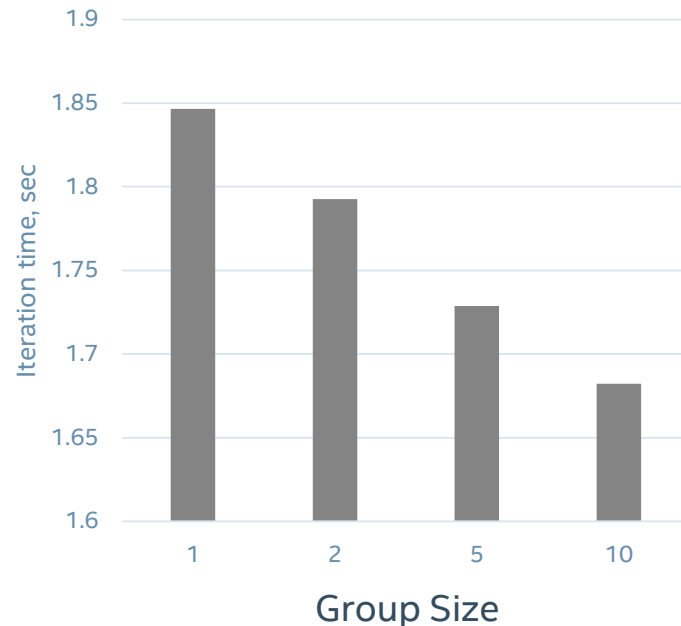
15% improvement

CIFAR, 32 nodes, 4cores,
EC2



24% improvement

ImageNet, 64 nodes, 16 cores



10% improvement

CONCLUSION

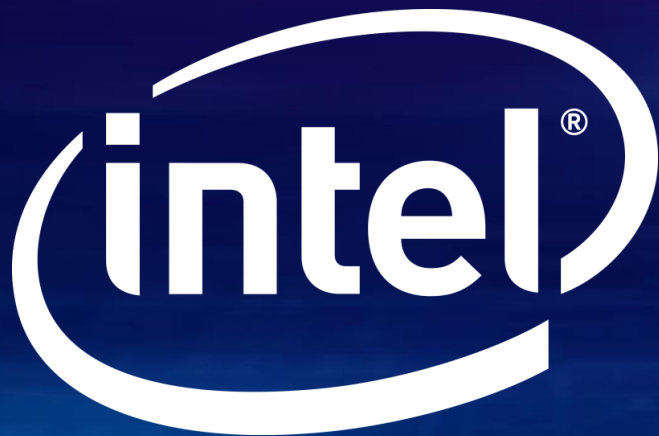
- **Deep Learning Spark jobs are somewhat unique**
 - **Heavy master node load for large model parameter update**
 - **Relatively short execution tasks (for fast model conversion)**
 - **Scheduling/Comms sometimes takes ~50% of total task execution.**
- **Deep Learning tasks are uniquely suited for optimization**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

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer. No computer system can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit <http://www.intel.com/performance>.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.

The products described 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.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Intel, the Intel logo, Pentium, Celeron, Atom, Core, Xeon and others are trademarks of Intel Corporation in the U.S. and/or other countries.

*Other names and brands may be claimed as the property of others.

© 2016 Intel Corporation.