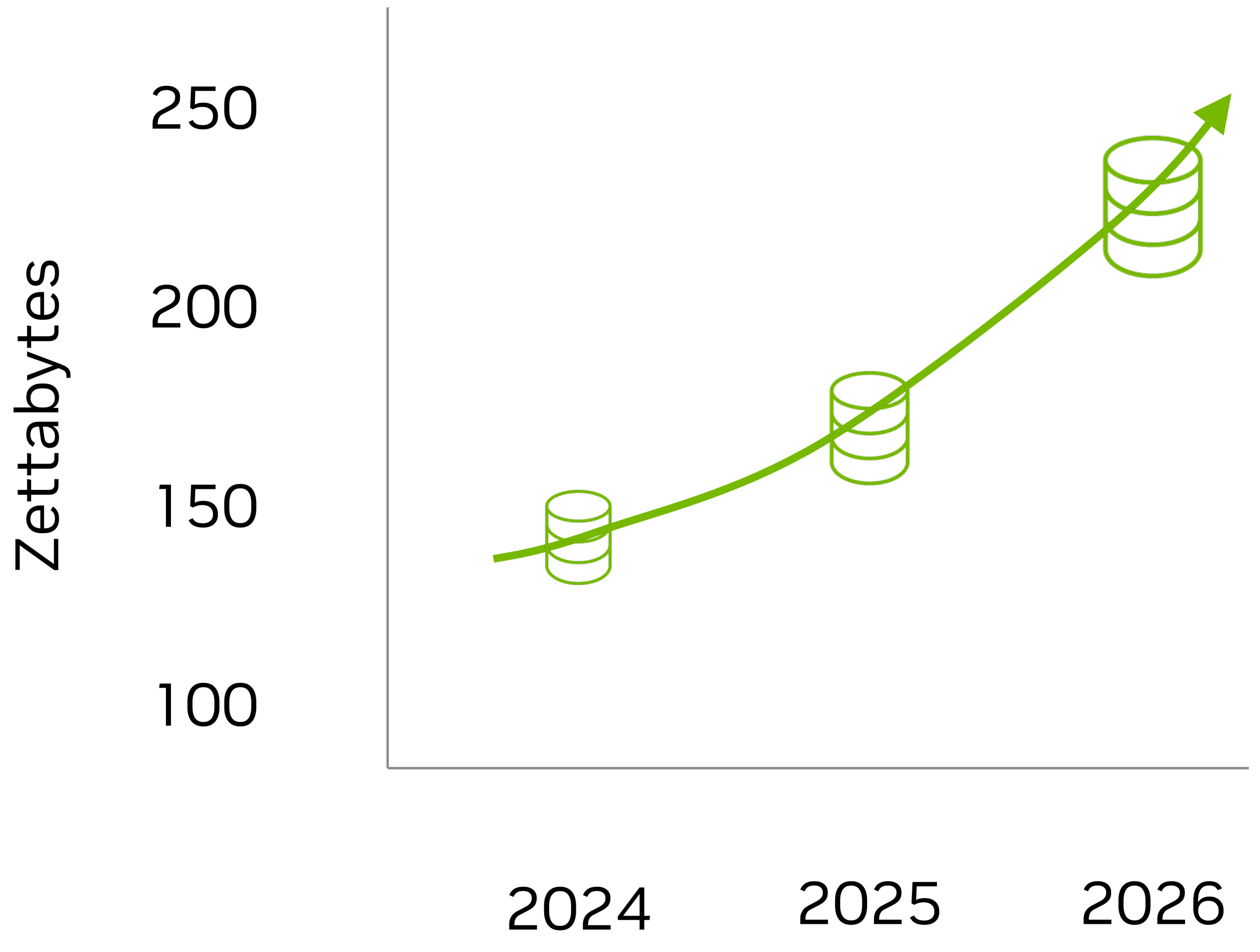
Accelerate ETL and Machine Learning in Apache Spark

Erik Ordentlich, Sameer Raheja | GTC | March 19, 2024

RAPIDS Spark

- Data Processing Challenges
- RAPIDS Accelerator for Apache Spark Data Processing
- RAPIDS Accelerator for Apache Spark ML
- Additional Information

Data Processing Compute Challenges



Data Growth

221 ZB of data by 2026

Scaling Compute

Moore’s law has slowed

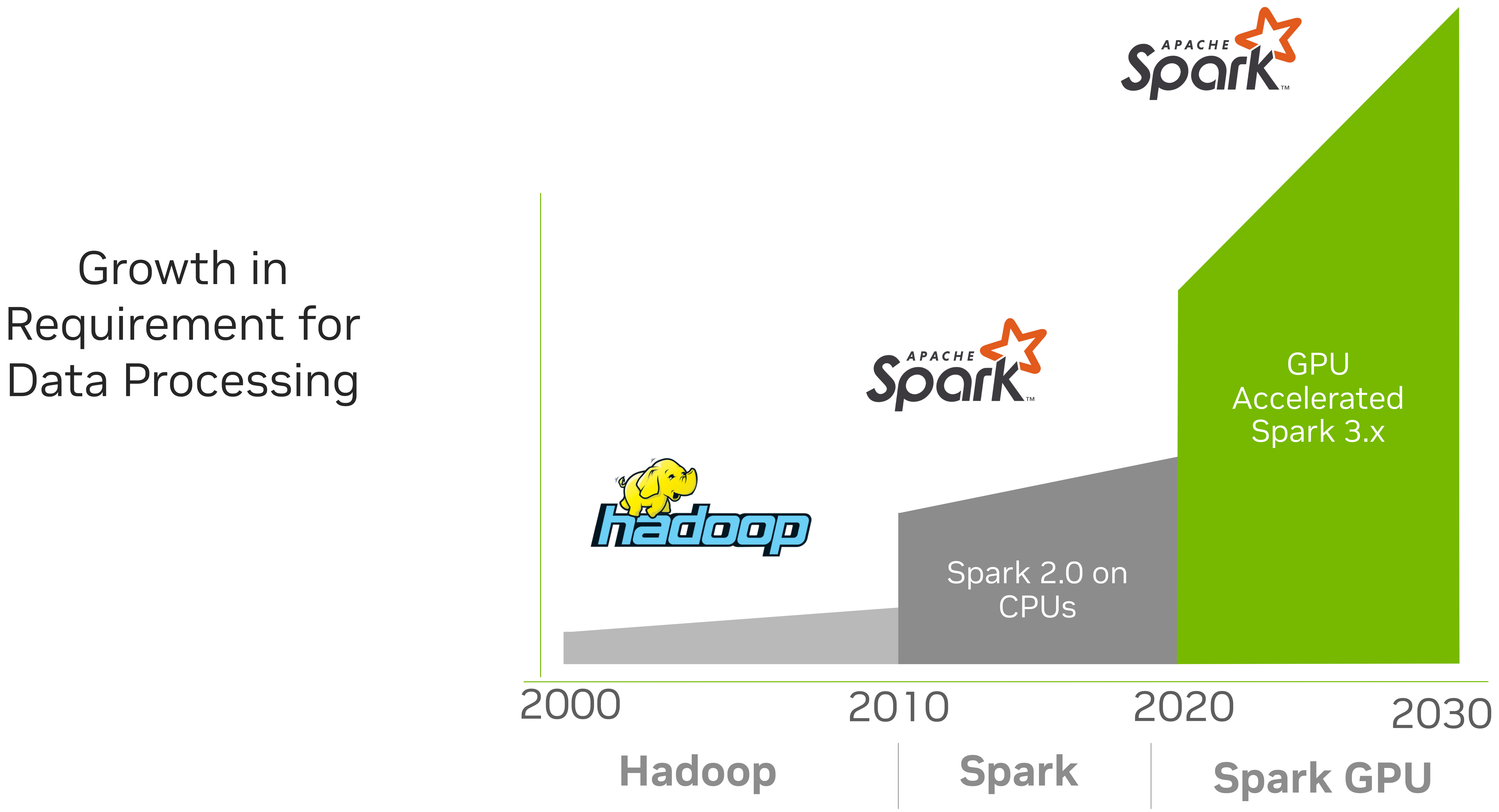
Power Consumption

Data centers account for 2% of the total US electricity use

Sources: IDC, Global DataSphere Forecast, 2022-2026: Enterprise Organizations Driving Most of the Growth, May 2022
https://www.energy.gov/eere/buildings/data-centers-and-servers

Scaling ETL Processing With Apache Spark with GPUs

RAPIDS Accelerator for Apache Spark



NVIDIA.

NVIDIA RAPIDS Accelerator

Key technologies for GPU acceleration

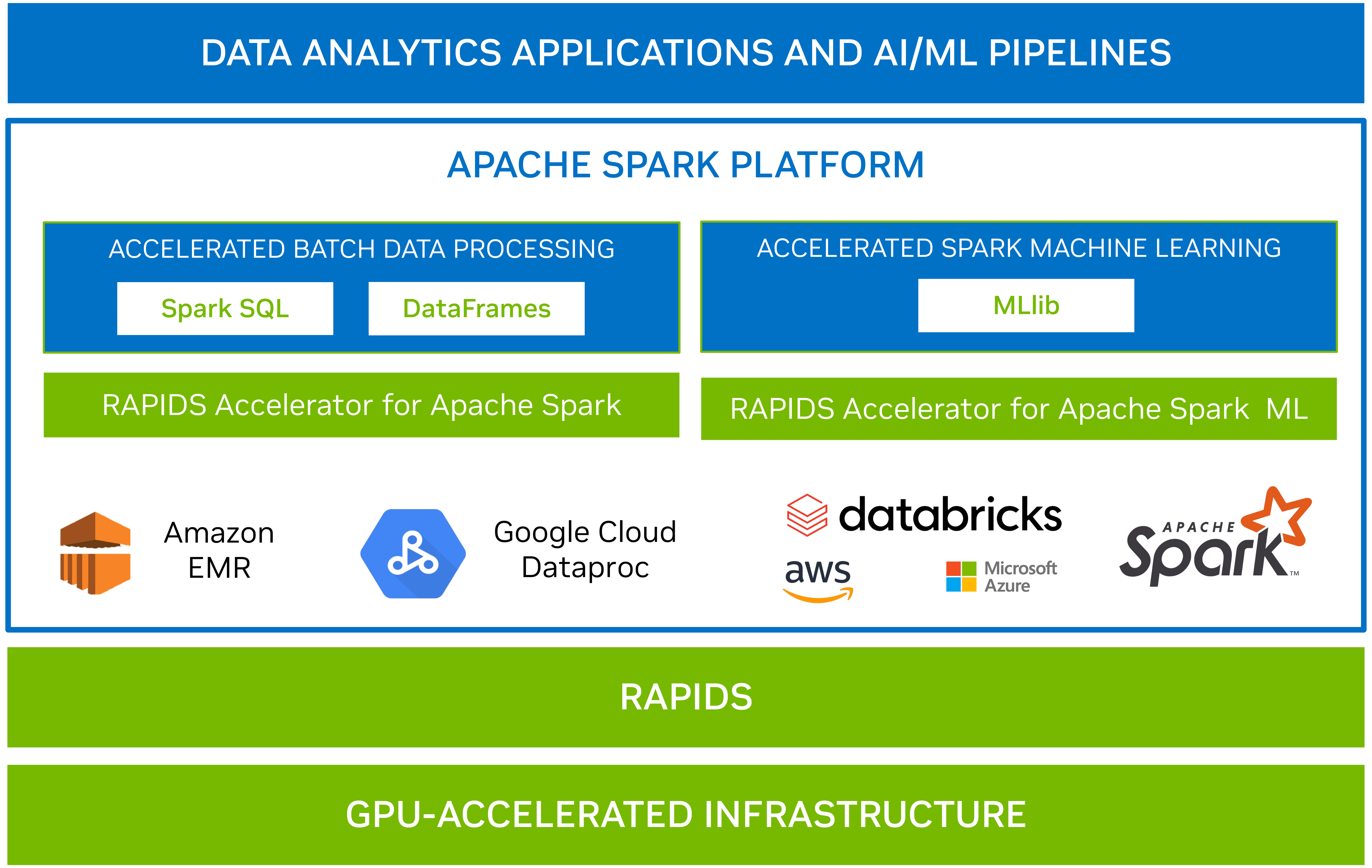
How it works

- Operates as a software plugin to popular Apache Spark platforms
  - Automatically accelerates supported operations
  - Requires no code changes
- Operations accelerated
  - Spark SQL
  - DataFrame
- Works with Spark standalone, YARN clusters, Kubernetes clusters

Key Spark 3 innovations

Columnar processing support in the Catalyst query optimizer – allows efficient GPU acceleration

GPU-aware scheduling of executors with a specified number of GPUs and how many GPUs for each task



RAPIDS Spark

- Data Processing Challenges
- RAPIDS Spark Data Processing
- RAPIDS Spark ML
- Additional Information

No Query Changes

- Add jar to classpath and set spark.plugins config
- Same SQL and DataFrame code
- Compatible with PySpark, SparkR, Java, Scala and other DataFrame-based APIs
- Seamless fallback to CPU for unsupported operations

```
spark.sql( """

    SELECT
        o_order_priority
        count(*) as order_count
    FROM
        orders
    WHERE
        o_orderdate >= DATE '1993-07-01'
        AND o_orderdate < DATE '1993-07-01' +
interval '3' month
        AND EXISTS (
            SELECT
                *
            FROM lineitem
            WHERE
                l_orderkey = o_orderkey
                AND l_commitdate < l_receiptdate
          )
    GROUP BY
        o_orderpriority ORDER BY o_orderpriority

""" ).show()
```

NVIDIA Decision Support Benchmark

NVIDIA Decision Support (NDS) is our adaptation of the TPC-DS benchmark often used by Spark customers and providers

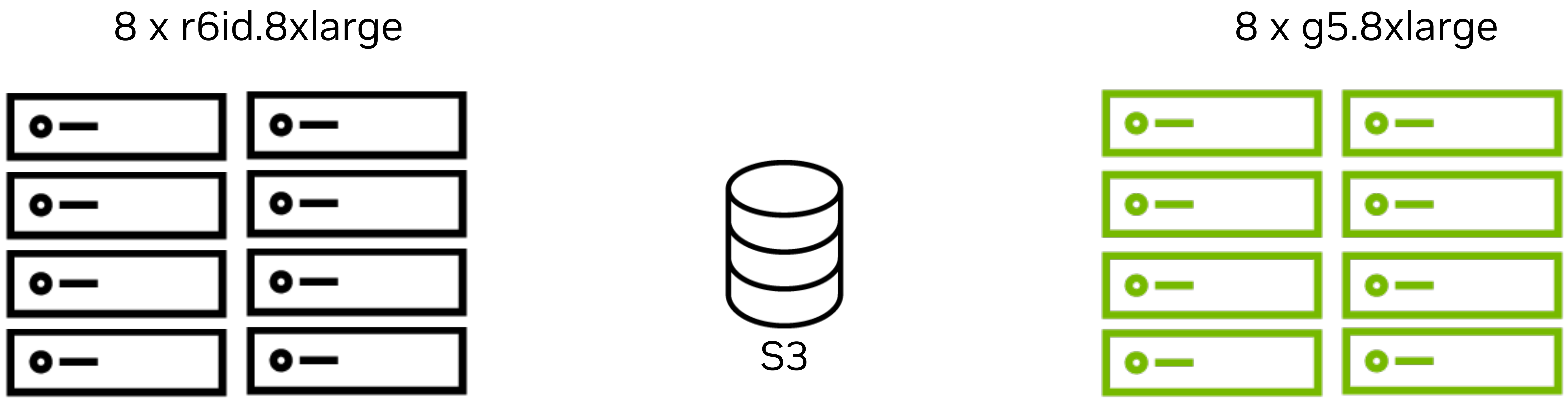
NDS consists of the same 100+ SQL queries as the industry standard benchmark but has modified parts for execution scripts.

The NDS benchmark is derived from the TPC-DS benchmark and as such is not comparable to published TPC-DS results, as the NDS results do not comply with the TPC-DS Specification

https://github.com/nvidia/spark-rapids-benchmarks

AWS EC2 cluster

Parquet data, scale factor 3k, stored on S3



	CPU Cores	CPU Mem (GB)	Network BW (Gbps)	Storage	GPU	On Demand $ Cost / Hr
r6id.8xlarge	32	256	12.5	1900GB local SSD		$2.419
g5.8xlarge	32	128	25	900GB local SSD	A10	$2.448

Source: https://aws.amazon.com/ec2/pricing/on-demand/

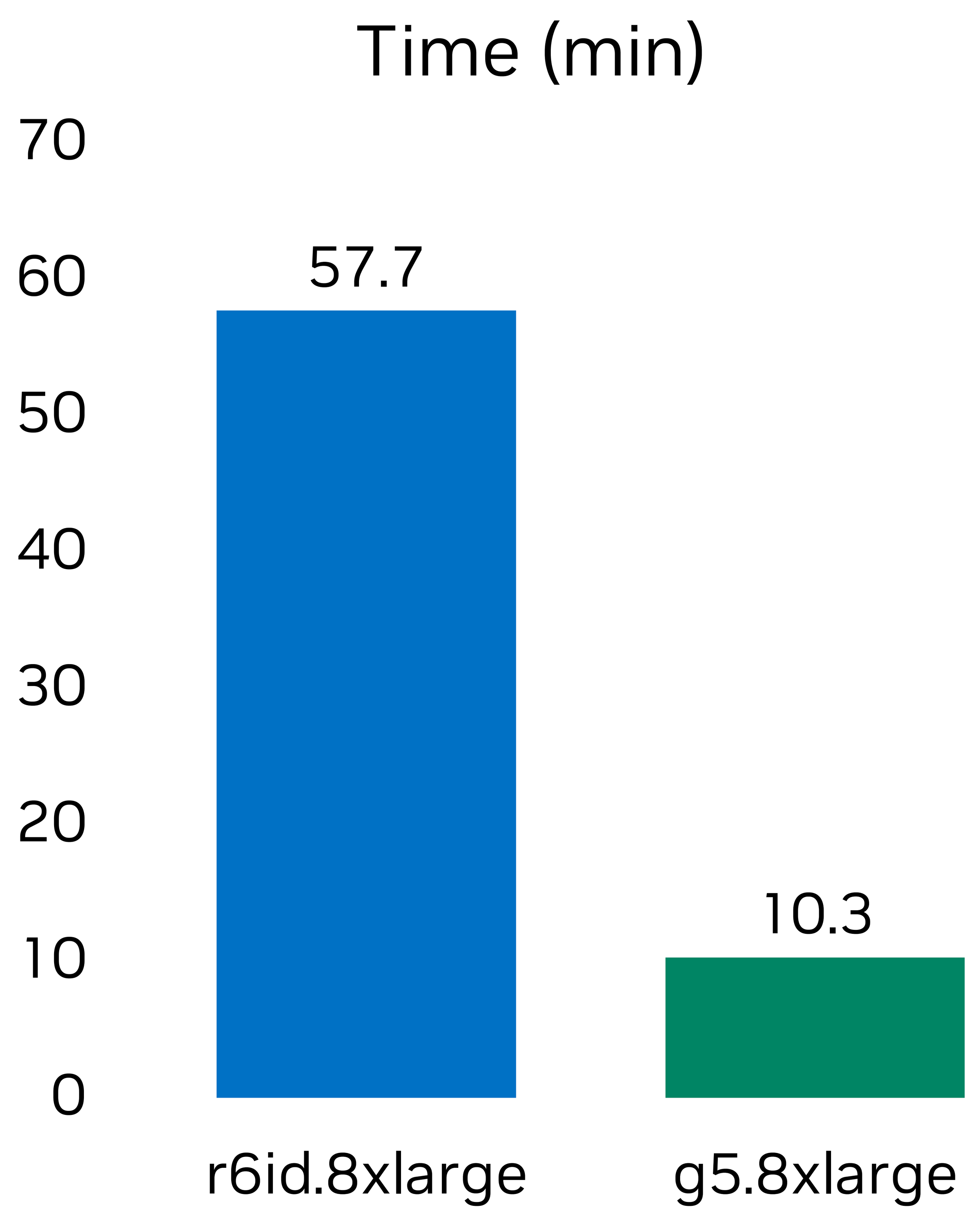
AWS EC2

Configurations

	CPU	GPU	Config type
spark.driver.memory	16G	16G	Resource
spark.executor.cores	16	16	
spark.executor.instances	16	8	
spark.executor.memory	64G	64G	
spark.rapids.filecache.enabled		true	
spark.executor.resource.gpu.amount		1	
spark.task.resource.gpu.amount		0.0625	
spark.scheduler.minRegisteredResourcesRatio	1.0	1.0	Scheduling
spark.locality.wait	0s	0s	
spark.sql.files.maxPartitionBytes	128mb (default)	2GB	Shuffle
spark.shuffle.manager		com.nvidia.spark.rapids.spark341.RapidsShuffleManager	
spark.rapids.shuffle.multiThreaded.{reader|writer}.threads		32	
spark.rapids.sql.multiThreadedRead.numThreads		100	
spark.plugins		com.nvidia.spark.SQLPlugin	GPU
spark.rapids.memory.host.spillStorageSize		16G	
spark.rapids.memory.pinnedPool.size		8G	
spark.rapids.sql.concurrentGpuTasks		3	

NVIDIA Decision Support Benchmark 3TB, AWS EC2

Apache Spark 3.4.1, RAPIDS Spark release 24.04



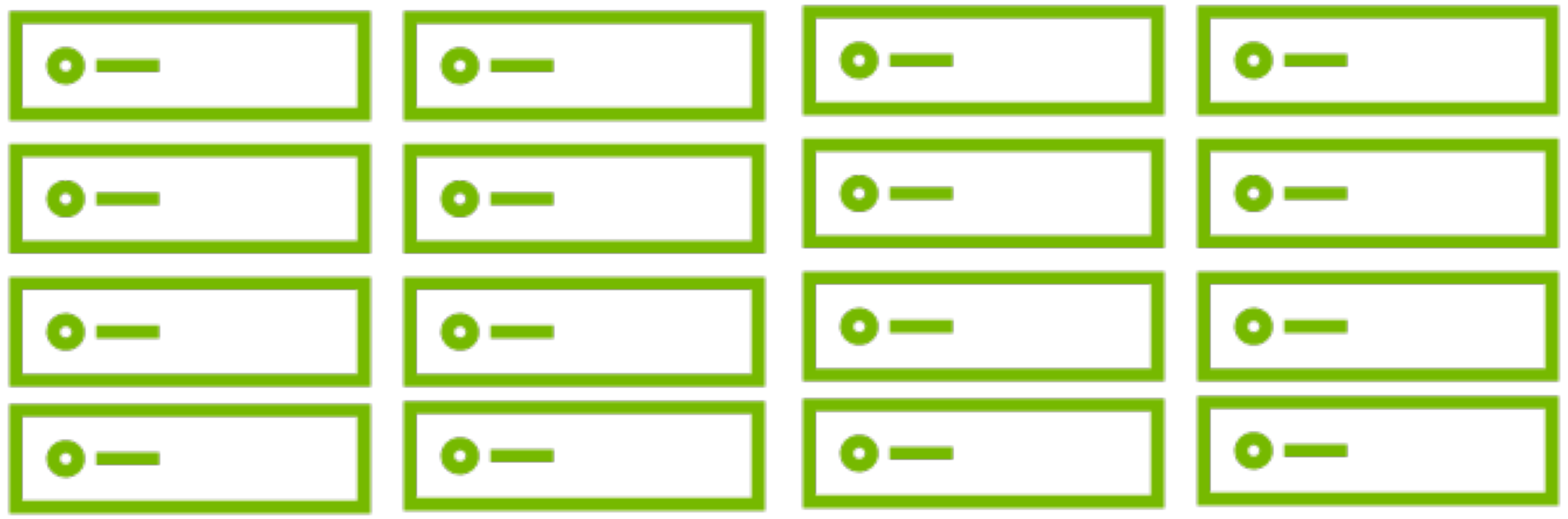
5.5x faster

80% cost savings

Grace Hopper (GH200) 16 Node Cluster

Parquet data stored on HDFS

16 x Grace Hopper Nodes



	CPU Cores	CPU Mem (GB)	Network BW (Gbps)	Storage	GPU	Retail Price / node
Quanta GH200	72	512	100	4 x 3.8TB local SSD	H100	$45763

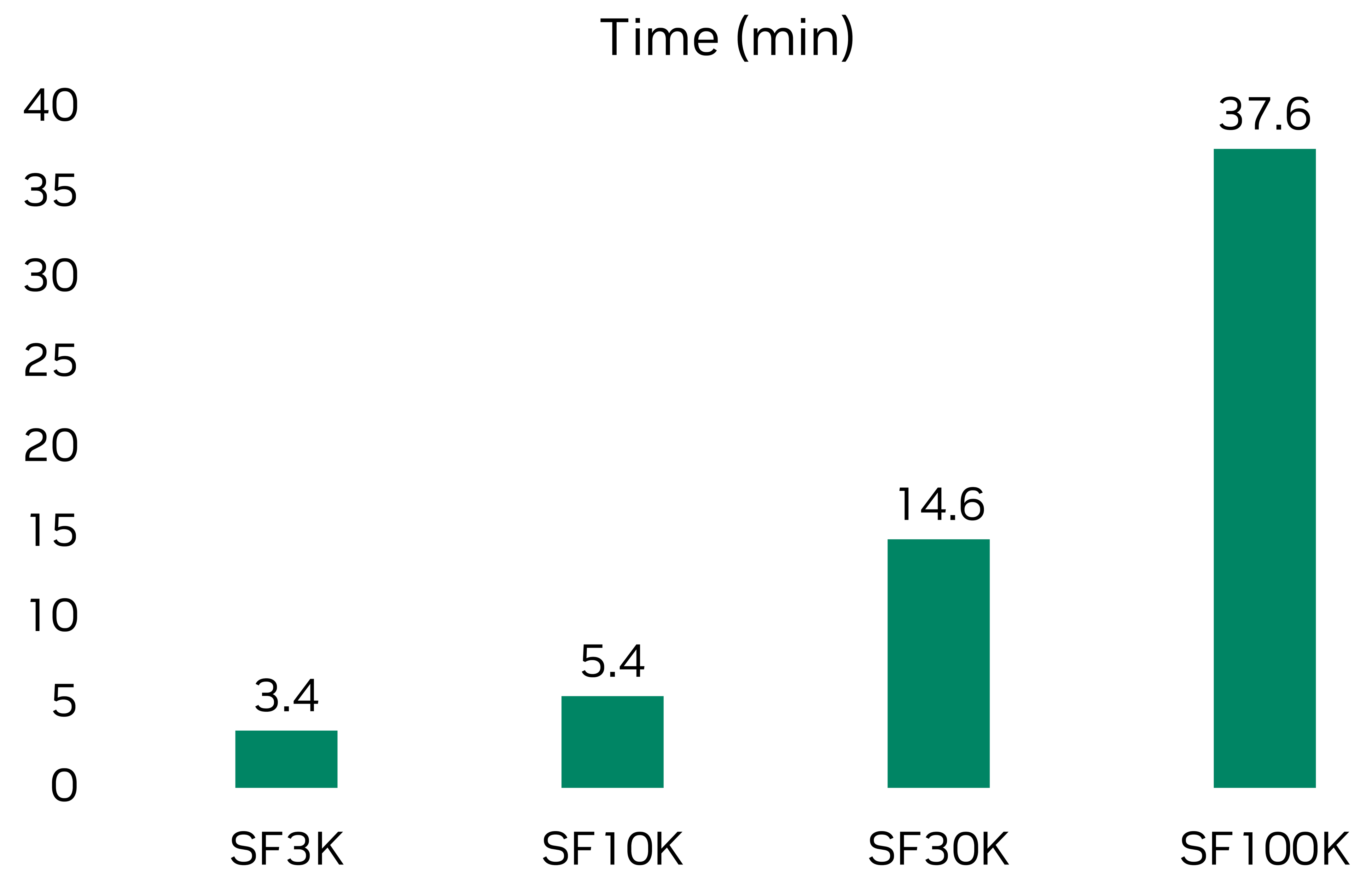
Grace Hopper

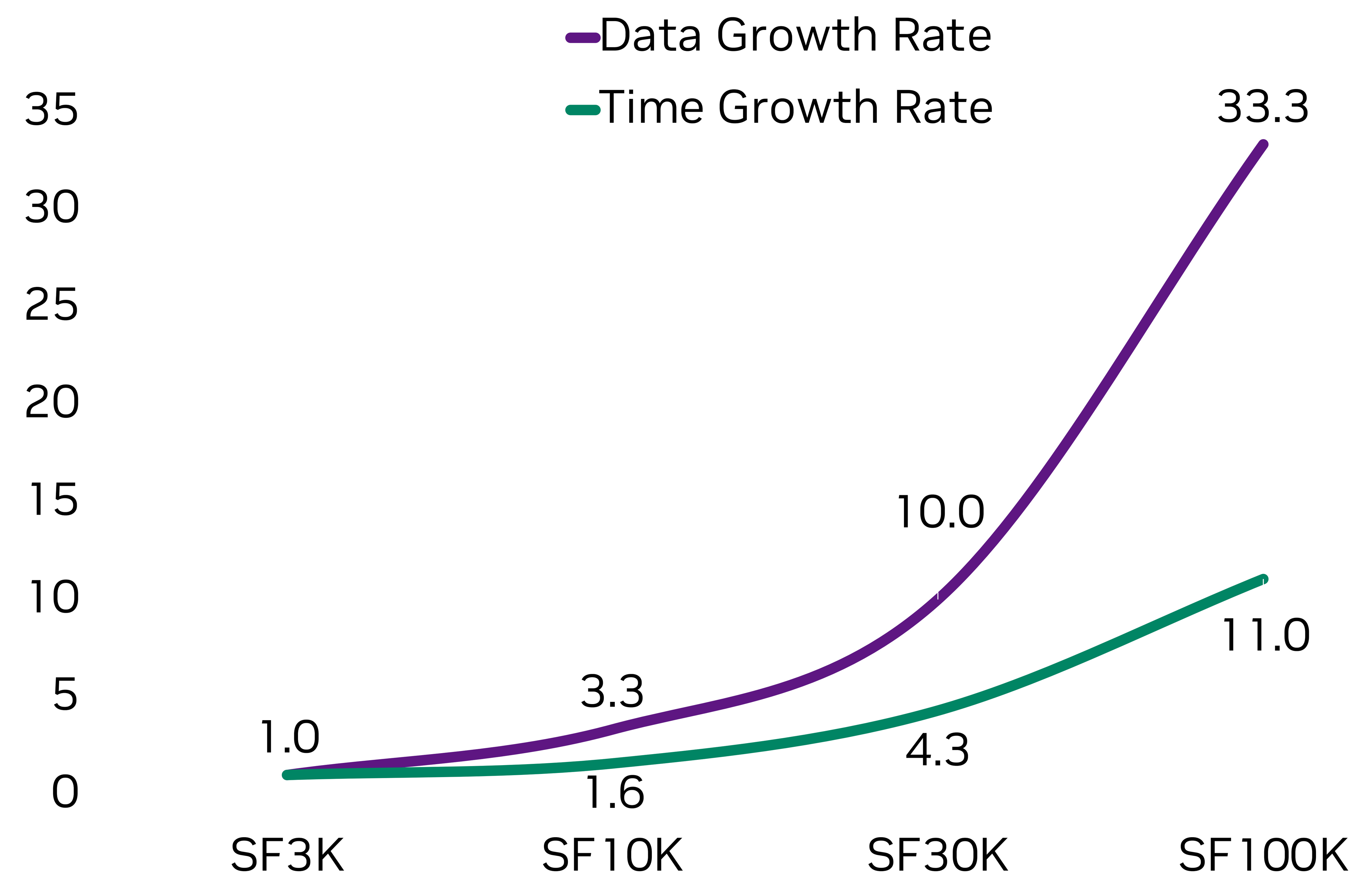
Configurations

	GPU	Config type
spark.driver.memory	50G	Resource
spark.driver.maxResultSize	2G	
spark.executor.cores	16	
spark.executor.memory	16G	
spark.rapids.filecache.enabled	true	
spark.executor.resource.gpu.amount	1	
spark.task.resource.gpu.amount	0.0625	
		Scheduling
spark.locality.wait	0s	
spark.sql.files.maxPartitionBytes	2GB	Shuffle
spark.shuffle.manager	com.nvidia.spark.rapids.spark341.RapidsShuffleManager	
spark.rapids.shuffle.multiThreaded.{reader|writer}.threads	32	
spark.plugins	com.nvidia.spark.SQLPlugin	GPU
spark.rapids.memory.host.spillStorageSize	32G	
spark.rapids.memory.pinnedPool.size	8G	
spark.rapids.sql.concurrentGpuTasks	4	

Grace Hopper (GH200) 16 Node Cluster

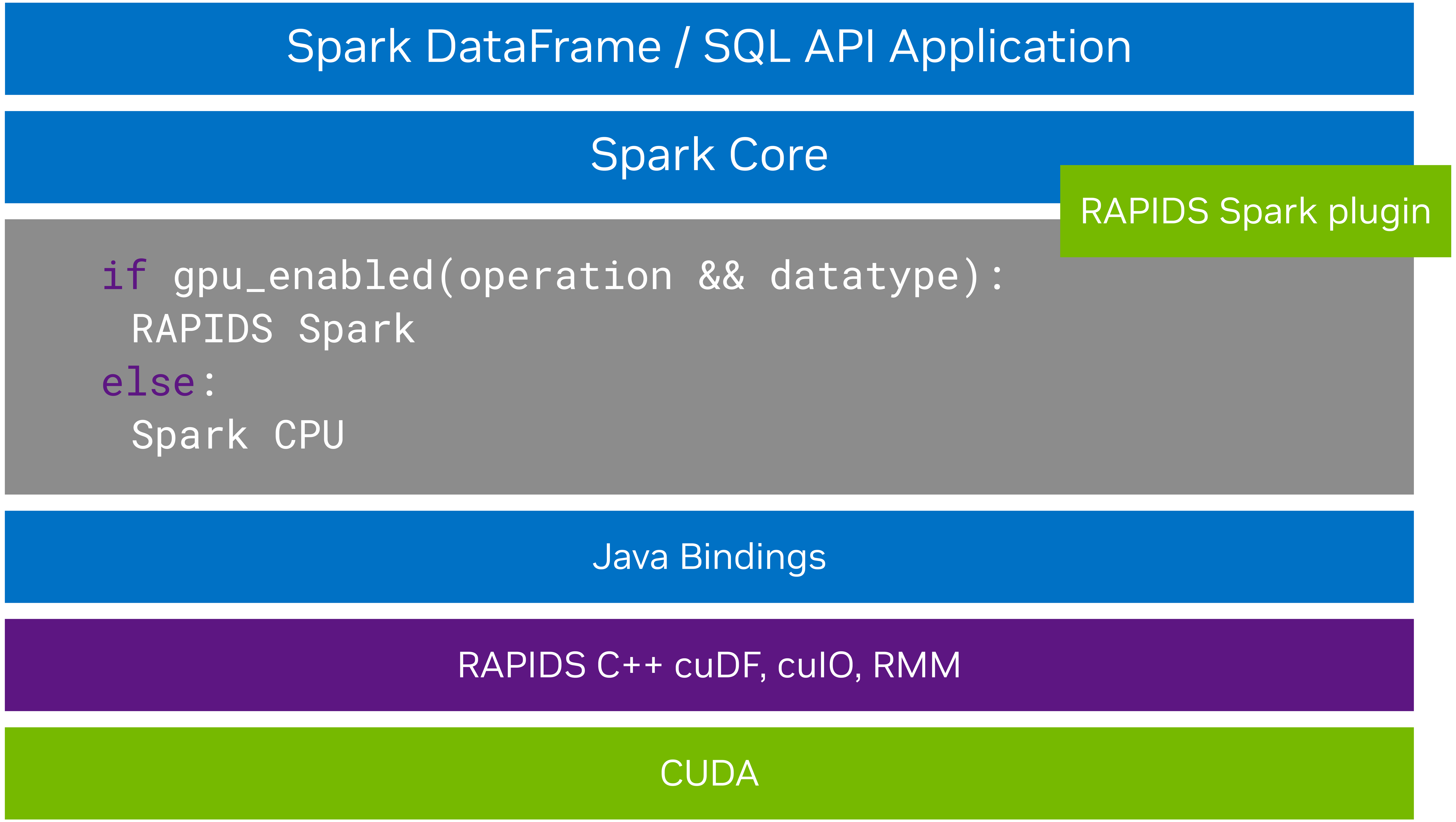
RAPIDS Spark 24.04





RAPIDS Accelerator for Apache Spark

Spark Plugin for GPU Acceleration



Spark SQL & DataFrame Query Execution

RAPIDS Spark Qualification Tool

Predicting the benefit of Spark + GPUs

spark-rapids-user-tools 24.2.1

```
pip install spark-rapids-user-tools
```

Latest version

Released: Mar 14, 2024

A simple wrapper process around cloud service providers to run tools for the RAPIDS Accelerator for Apache Spark.

Navigation

- Project description
- Release history
- Download files

Statistics

View statistics for this project via Libraries.io, or by using our public dataset on Google BigQuery

Meta

License: Apache Software License

Author: NVIDIA Corporation

Requires: Python >=3.8

Project description

spark-rapids-user-tools

User tools to help with the adoption, installation, execution, and tuning of RAPIDS Accelerator for Apache Spark.

The wrapper improves end-user experience within the following dimensions:

1. **Qualification**: Educate the CPU customer on the cost savings and acceleration potential of RAPIDS Accelerator for Apache Spark. The output shows a list of apps recommended for RAPIDS Accelerator for Apache Spark with estimated savings and speed-up.
2. **Bootstrap**: Provide optimized RAPIDS Accelerator for Apache Spark configs based on GPU cluster shape. The output shows updated Spark config settings on driver node.
3. **Tuning**: Tune RAPIDS Accelerator for Apache Spark configs based on initial job run leveraging Spark event logs. The output shows recommended per-app RAPIDS Accelerator for Apache Spark config settings.
4. **Diagnostics**: Run diagnostic functions to validate the Dataproc with RAPIDS Accelerator for Apache Spark environment to make sure the cluster is healthy and ready for Spark jobs.

Getting started

Apache Spark Ecosystem

Supported Distributions

Open-source	Cloud			On-prem
Apache Spark 3+ Community Release	Amazon EMR	Databricks	Google Cloud Dataproc	Cloudera CDP

Improvements Over the Last Year

- Reliability
  - Spill framework to reduce OOM issues to minimize OOM or GPU specific config changes
  - OOM retry framework for automatic OOM handling in memory-intensive operators
- Performance
  - Dynamic repartitioning in large/skewed hash joins
  - File caching
  - Improved I/O and larger chunk handling for Parquet
- Usability
  - Tooling support on Azure and AWS Databricks, Google Dataproc and AWS EMR
- Scaling to 100s of TB and beyond
- ARM support
- JSON handling improvements
- Support for Delta Lake

Roadmap

- Blackwell GPU will have hardware decompression for snappy and zstd
- Qualification tool with an ML model to improve prediction
- Support for Apache Hudi and Apache Iceberg
- Performance improvements reading from cloud object stores
- Scaling improvements for GPU hardware

RAPIDS Spark

- Data Processing Challenges
- RAPIDS Spark Data Processing
- RAPIDS Spark ML
- Additional Information

NVIDIA RAPIDS Accelerator

Key technologies for GPU acceleration

DATA ANALYTICS APPLICATIONS AND AI/ML PIPELINES

APACHE SPARK PLATFORM

ACCELERATED BATCH DATA PROCESSING

Spark SQL | DataFrames

RAPIDS Accelerator for Apache Spark

ACCELERATED SPARK MACHINE LEARNING

MLlib

RAPIDS Accelerator for Apache Spark ML

Amazon EMR | Google Cloud Dataproc | databricks | aws | Microsoft Azure | Apache Spark

RAPIDS

GPU-ACCELERATED INFRASTRUCTURE

NVIDIA / spark-rapids-ml

RAPIDS Spark ML

Motivation

```
spark.sql("SELECT * FROM range(10) where id > 7")

df.join(df2, 'name').select(df.weight, df2.height)

pyspark.ml.clustering.KMeans().fit(df)
```

RAPIDS Accelerator for Apache Spark

RAPIDS Accelerator for Apache Spark ML

NVIDIA RAPIDS Accelerator

Package import change

- Compatible with pyspark.ml DataFrame APIs
- Requires no application code change
- Package import change

```
from pyspark.ml.clustering import Kmeans

kmeans_estm = KMeans()\
.setK(100)\
.setFeaturesCol("features")\
.setMaxIter(30)

kmeans_model =
kmeans_estm.fit(pyspark_data_frame)

kmeans_model.write().save("saved-model")

transformed =
kmeans_model.transform(pyspark_data_frame)
```

Package import change

- Compatible with pyspark.ml DataFrame APIs
- Requires no application code change
- Package import change for acceleration

```
from spark_rapids_ml.clustering import Kmeans

kmeans_estm = KMeans()\
.setK(100)\
.setFeaturesCol("features")\
.setMaxIter(30)

kmeans_model =
kmeans_estm.fit(pyspark_data_frame)

kmeans_model.write().save("saved-model")

transformed =
kmeans_model.transform(pyspark_data_frame)
```

Distributed cuML integration

PySpark MLlib API

PCA

KMeans

...

One-Task-Per-GPU scheduling on Spark DataFrame

cuML MNMG classes / Raft NCCL communication

GPU

GPU

GPU

Map PySpark MLlib API calls to cuML for supported Algos
Use PySpark APIs (task-per-gpu scheduling, repartition, mapInPandas, barrier, and broadcast) to setup the cuML Multi-Node Multi-GPU cluster

Process data on GPUs using NCCL for communication

RAPIDS Spark ML

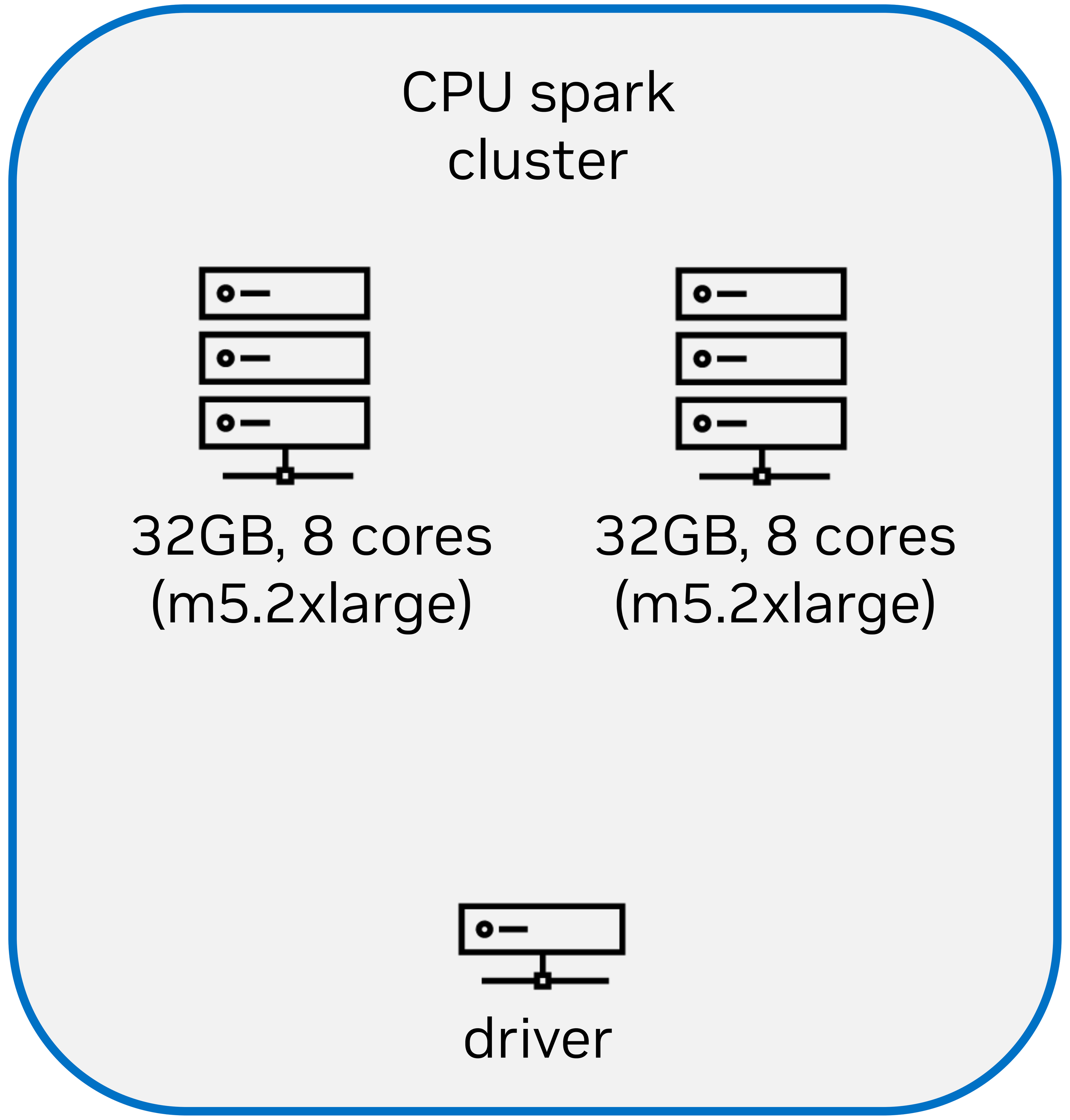
Supported Algorithms

In Spark MLlib only	In Spark MLlib and cuML	In cuML only
CrossValidator	K-Means	Exact k-NN
	Linear Regression	UMAP
	Logistic Regression	
	PCA	
	Random Forest Classifier	
	Random Forest Regressor	

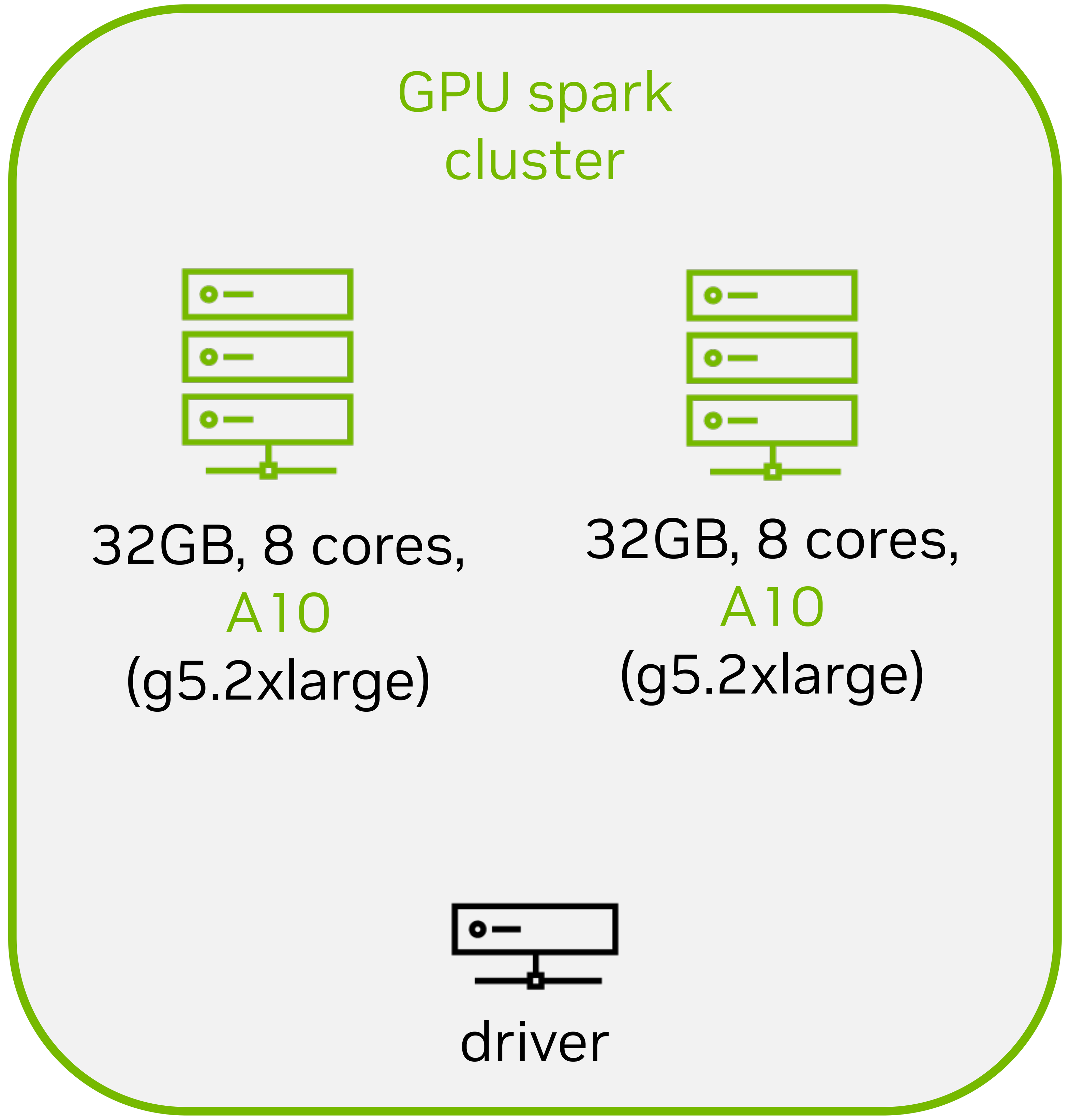
Example: MLlib-like API for GPU exact k-NN

```
>>> from spark_rapids_ml.knn import NearestNeighbors
>>> topk = 2
>>> gpu_knn = NearestNeighbors().setInputCol("features").setIdCol("id").setK(topk)
>>> gpu_model = gpu_knn.fit(data_df)
>>> (_, _, knn_df) = gpu_model.kneighbors(query_df)
>>> knnjoin_df = gpu_model.exactNearestNeighborsJoin(query_df, distCol="EuclideanDistance")
>>> knnjoin_df.show()
+---------------+---------------+-----------------+
|        item_df|       query_df|EuclideanDistance|
+---------------+---------------+-----------------+
|{1, [2.0, 2.0]}|{3, [1.0, 1.0]}|        1.4142135|
|{0, [1.0, 1.0]}|{3, [1.0, 1.0]}|              0.0|
|{2, [3.0, 3.0]}|{4, [3.0, 3.0]}|              0.0|
|{1, [2.0, 2.0]}|{4, [3.0, 3.0]}|        1.4142135|
+---------------+---------------+-----------------+
```

Microbenchmarking

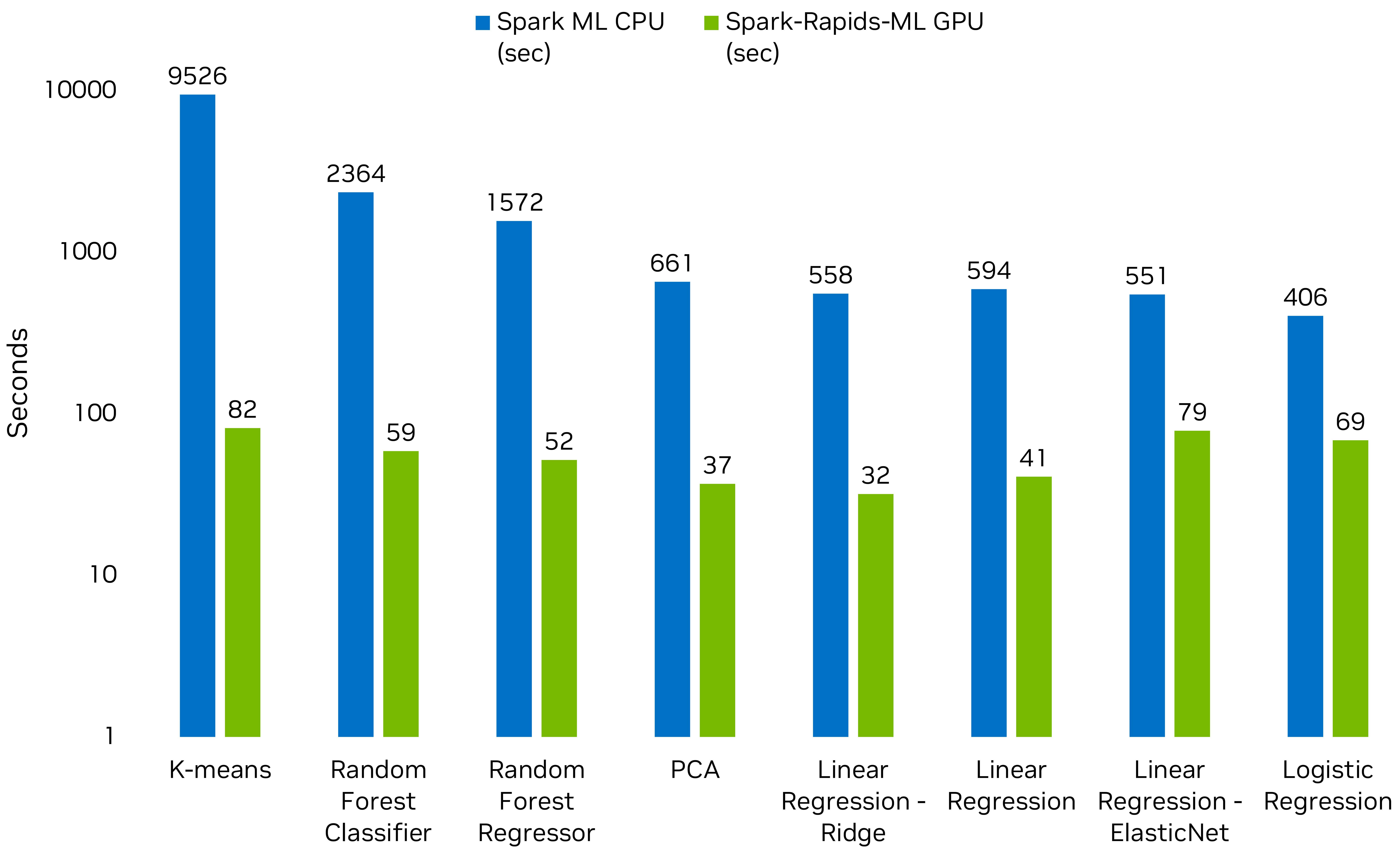


- **Environment**
  - Databricks AWS hosted Spark
- **Workload & Data**
  - estimator.fit(data_df) [i.e. training]
  - data_df read from Parquet format in AWS S3
  - Compute intensive synthetic workloads:
    - 1 million rows
    - 3000 dimensional vectors
  - Data available in S3 public bucket.

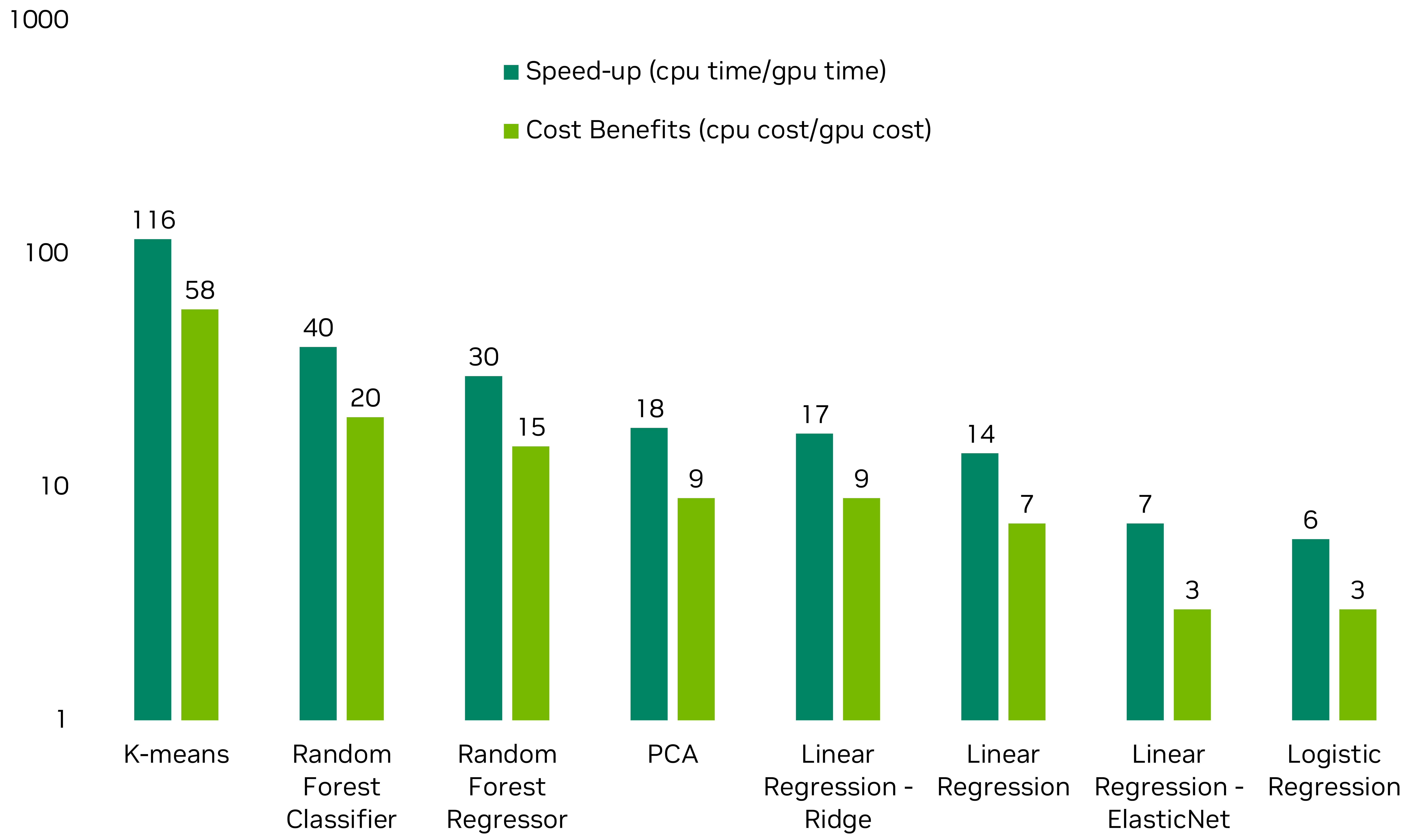


- Instructions and scripts to reproduce: https://github.com/NVIDIA/spark-rapids-ml/tree/main/python/benchmark#databricks
- [Repo also has instructions for GCP Dataproc and AWS EMR]

Training/fit time: 6x-100x faster



Cost Benefits and Speedups



End-to-End Acceleration

Stage Level Scheduling

- ML training stage runs all tasks at the same time with one Task for each GPU
  - Required by cuML/NCCL layer
- ETL can benefit from multiple Tasks per GPU
- Stage level scheduling:
  - Allows different tasks per GPU on a per stage basis within the same Spark Job.
- GPU aware stage level scheduling

End-to-End Acceleration

Accelerated CrossValidator

- PySpark API compatibility allows all accelerated Algos to leverage PySpark's built in CrossValidator for hyper parameter tuning
- We can do better:

PySpark MLlib CrossValidator

RAPIDS Spark ML CrossValidator

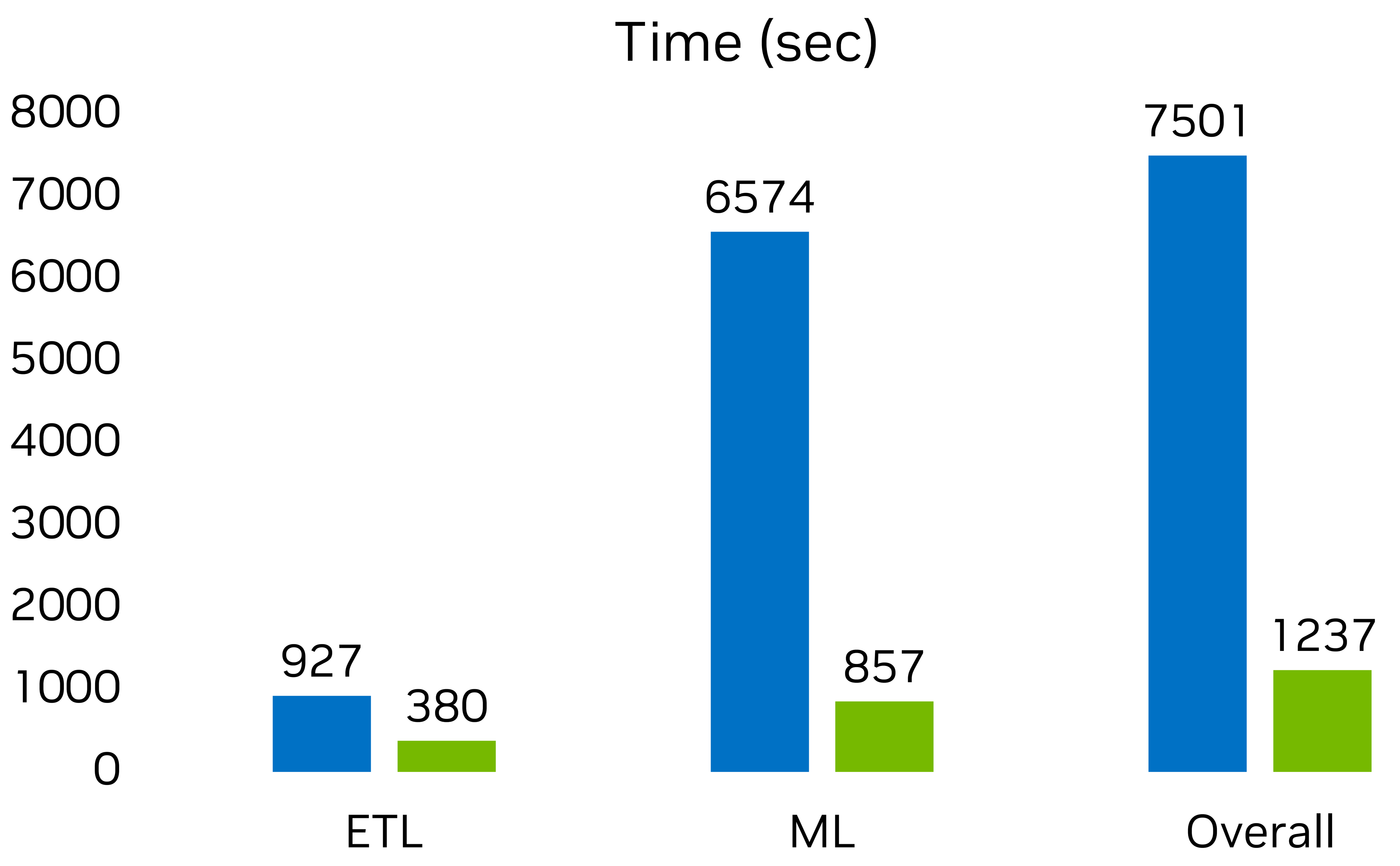
Fannie Mae Mortgage Benchmark

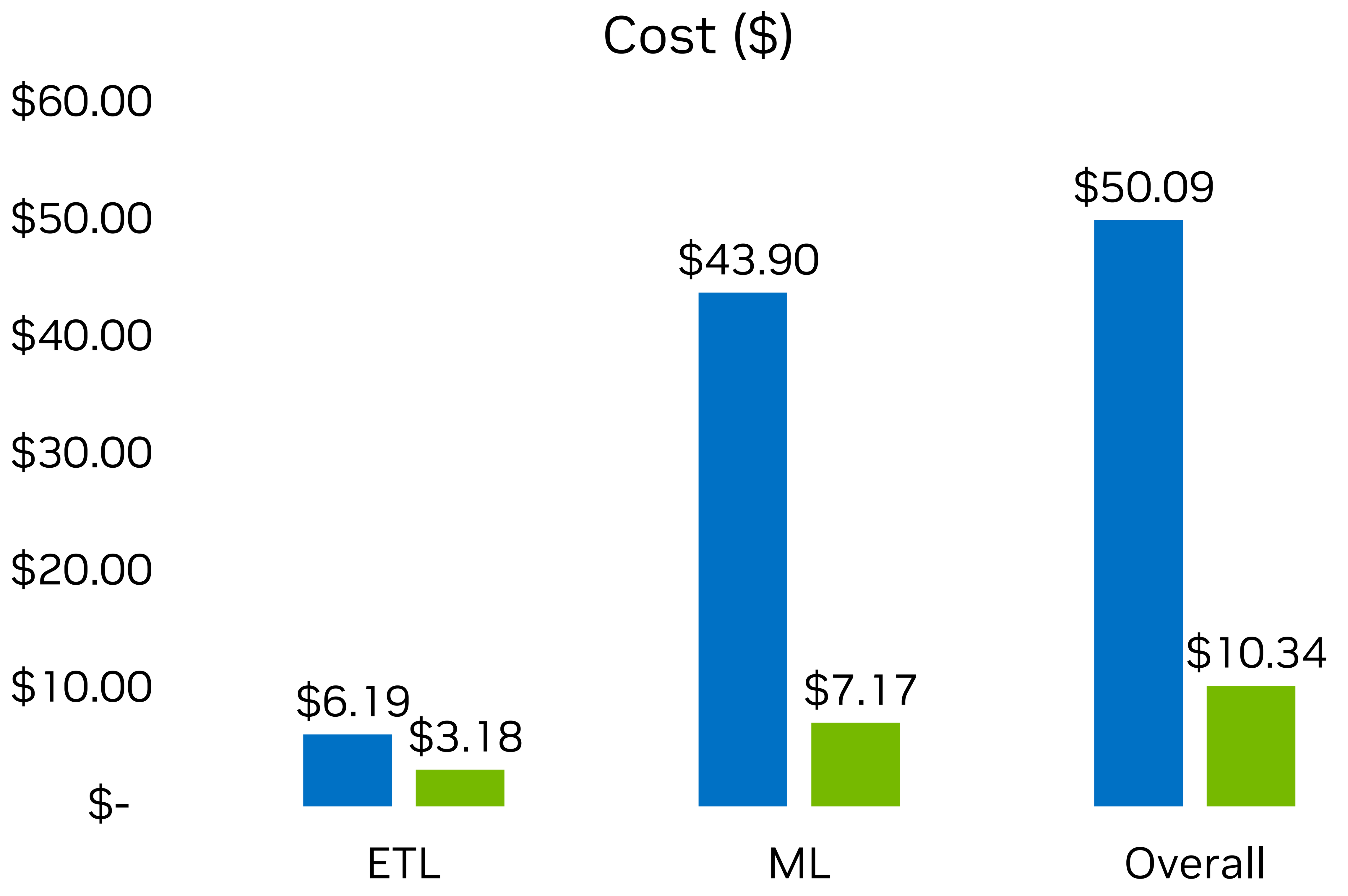
ETL + ML

- End-to-end ETL – ML workload:
  - ETL:
    - Starts with compressed Fannie Mae Single-Family Loan Performance Data ~ 800GB csv dataset converted to 26.8 GB compressed Parquet.
    - Feature engineering to 2.6 billion records by 27 feature dataset with loan delinquency as label. (as in this example: https://github.com/NVIDIA/spark-rapids-examples/.../MortgageETL.ipynb).
  - ML
    - Logistic regression with 3 fold cross validation wrt to log loss over 8 algo parameter choices.
  - Environment: Databricks AWS 16 node clusters:

Fannie Mae Mortgage Benchmark

ETL + ML





6.1x faster end-to-end

4.8x lower cost

Accuracy: GPU – CPU ave CV score < 0.004%

Roadmap

- Better out-of-core
- Blackwell and Grace-Hopper optimizations
- Spark APIs for more algorithms from cuML:
  - Batch approximate nearest neighbor vector search
  - DBSCAN clustering
- GPU optimized Pipelines
- Additional Spark MLlib algos

Broader GPU Accelerated ML/DL on Spark Ecosystem

Hugging Face

Introducing
Llama 2

dmlc XGBoost

TensorFlow

PyTorch

APACHE Spark™

NVIDIA RAPIDS

NVIDIA TRITON INFERENCE SERVER

• • •

NVIDIA TENSOR RT

NVIDIA CUDA and GPUS

RAPIDS Spark

- Data Processing Challenges
- RAPIDS Spark Data Processing
- RAPIDS Spark ML
- Additional Information

NVIDIA AI Enterprise

End to end software platform for AI and Data Science

MLOps | AI Applications

NVIDIA AI Enterprise

Infrastructure Management

Cloud Native Management and Orchestration
GPU Operator, Network Operator

Cluster Management
Base Command Manager Essentials

Infra Acceleration Libraries
Magnum IO, vGPU, CUDA

Application Frameworks

- **LLM** – NeMo
- **Speech AI** – Riva
- **Cybersecurity** – Morpheus
- **Medical Imaging** – Clara
- More

AI Development

Data Science / Prep
RAPIDS, RAPIDS Accelerator for Apache Spark

→ **Model Training and Customization**
NeMo, TAO, PyTorch, TensorFlow

→ **Optimize for Inference**
TensorRT, TensorRT-LLM

→ **Deploy at Scale**
Triton Inference Server

Cloud | Data Center | Workstations | Edge

Learn More!

PayPal: How PayPal Reduces Cloud Costs by up to 70% with Spark RAPIDS [S62506]
Wed 8:30-8:55

Baidu: From SQL to Chat: How to Revolutionize Enterprise Data Analysis with NVIDIA [S61622]
Wed 9:00-9:25

north.io: How AI and Accelerated Computing are Revolutionizing Oceanographic Data Processing [S61391]
Wed 3:00-3:25

Taboola: RAPIDS Accelerator for Apache Spark Propels Data Center Efficiency and Cost Savings [S62130]
Thu 10:00-10:25

Sessions

Reduce Apache Spark MLlib Costs with NVIDIA GPUs [CWE62407]
Wed 4:00-4:50

Cost Savings and Speedup with the RAPIDS Accelerator for Apache Spark [CWE62404]
Thu 9:00-9:50

Connect with Experts

Accelerating Data Analytics on GPUs with the RAPIDS Accelerator for Apache Spark [DLIT61196]
Wed 8:00-9:40

Tutorial

For More Information

spark-rapids-support@nvidia.com

https://docs.nvidia.com/spark-rapids

https://nvidia.github.io/spark-rapids

https://github.com/NVIDIA/spark-rapids-ml

https://nvidia.github.io/spark-rapids-ml

NVIDIA