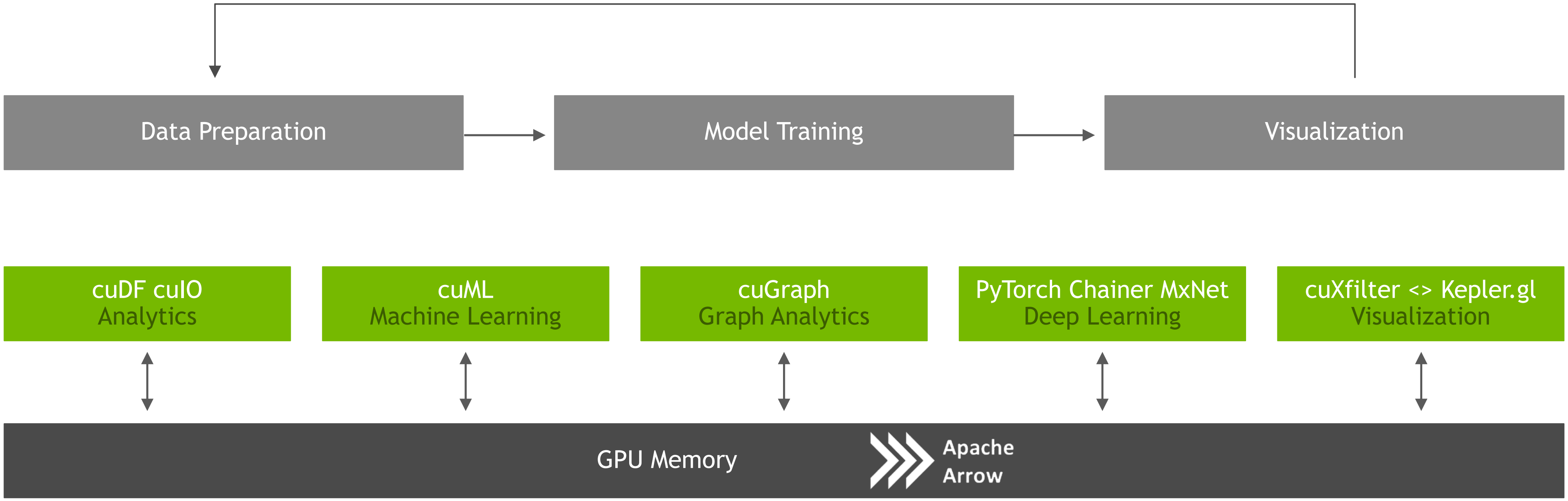
RAPIDS: PLATFORM INSIDE AND OUT

Joshua Patterson 3-19-2019

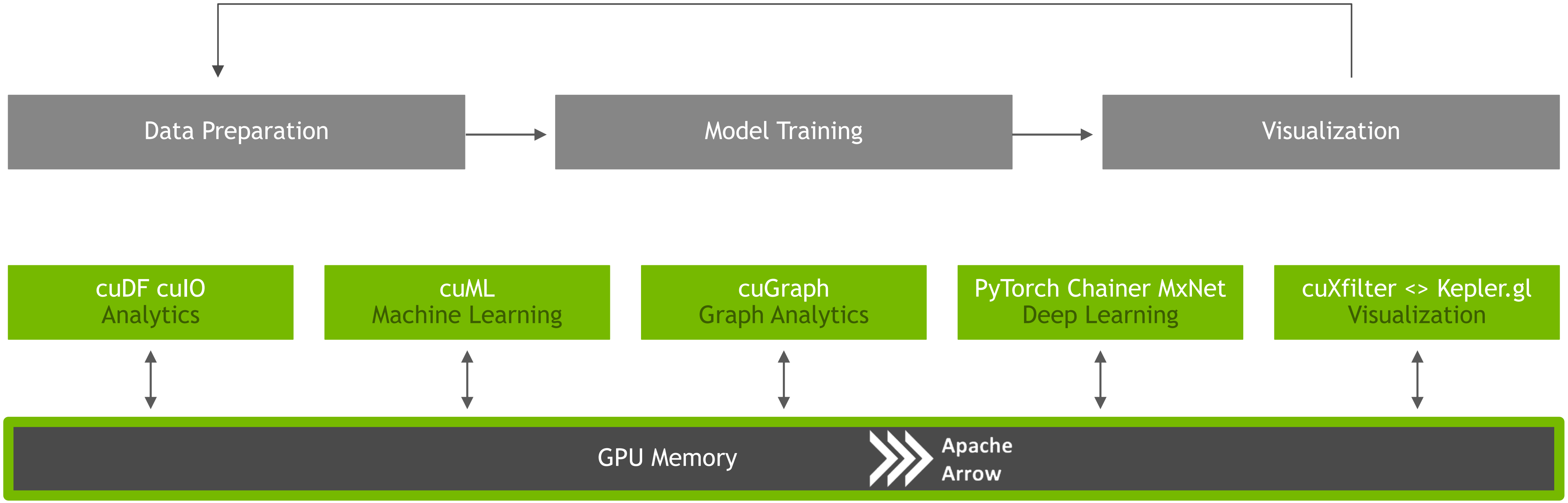
RAPIDS

End to End Accelerate GPU Data Science



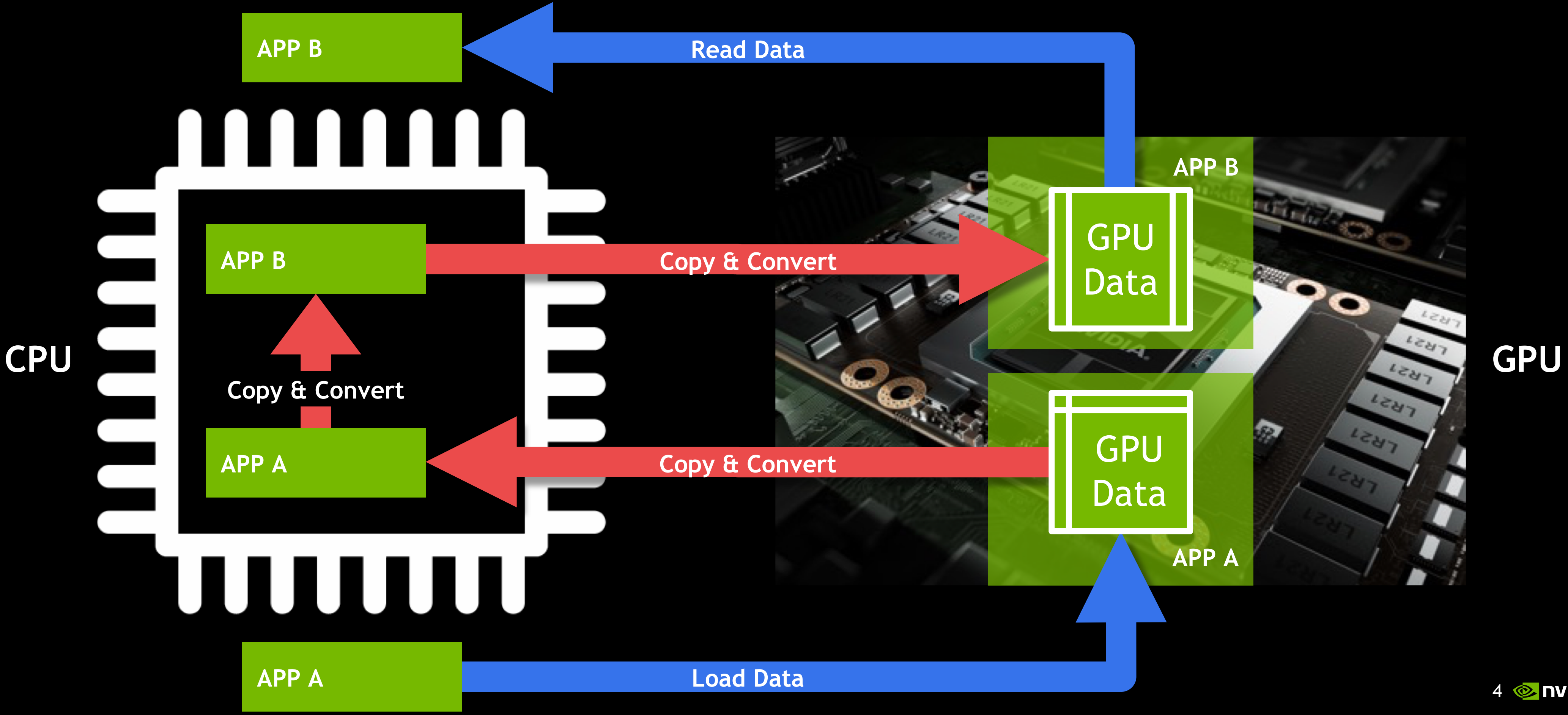
RAPIDS

End to End Accelerate GPU Data Science



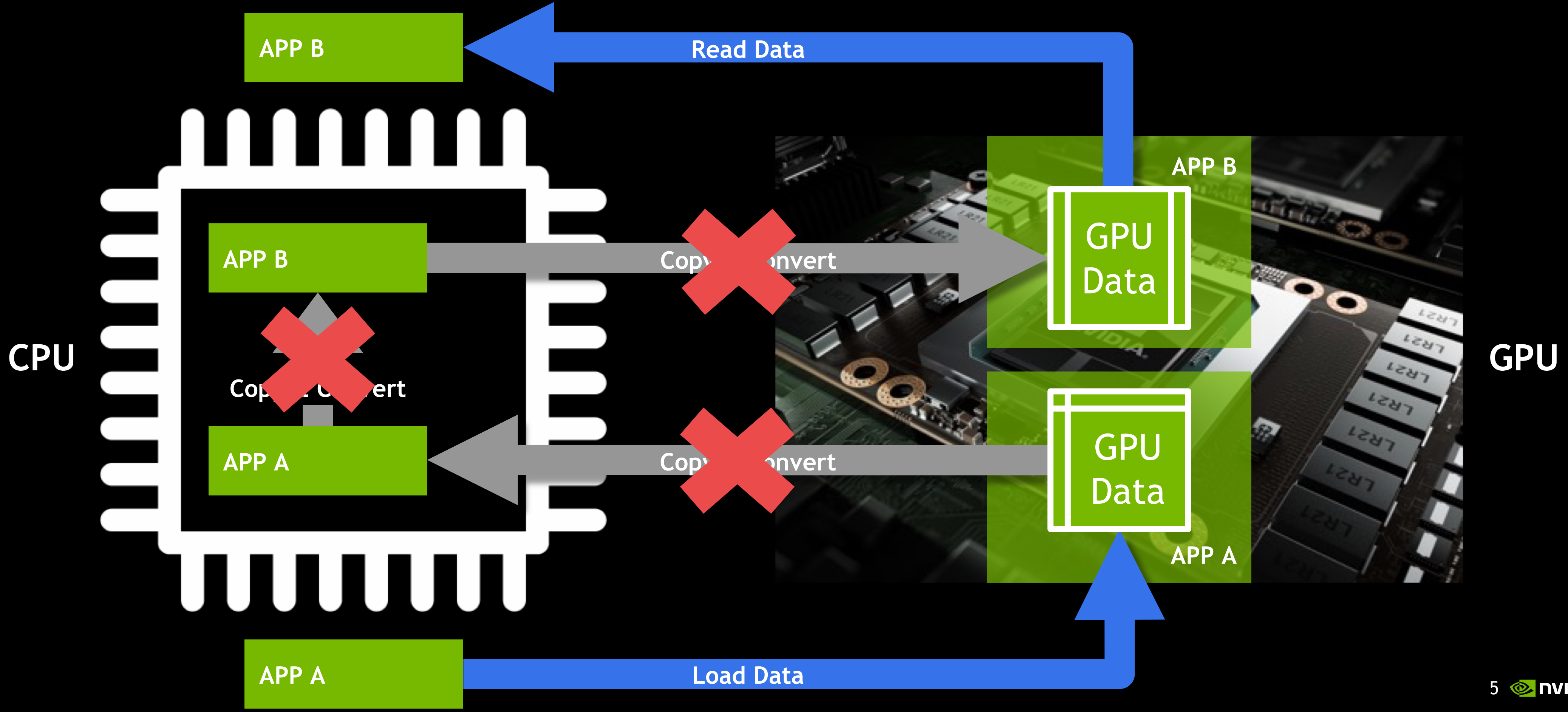
DATA MOVEMENT AND TRANSFORMATION

The bane of productivity and performance

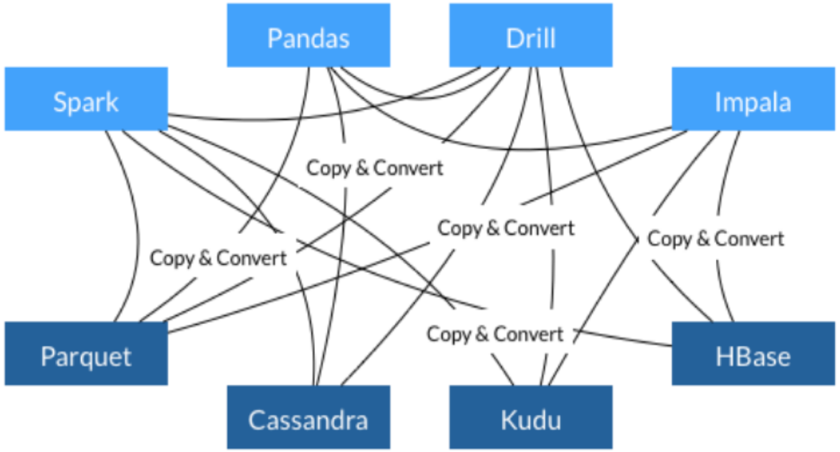


DATA MOVEMENT AND TRANSFORMATION

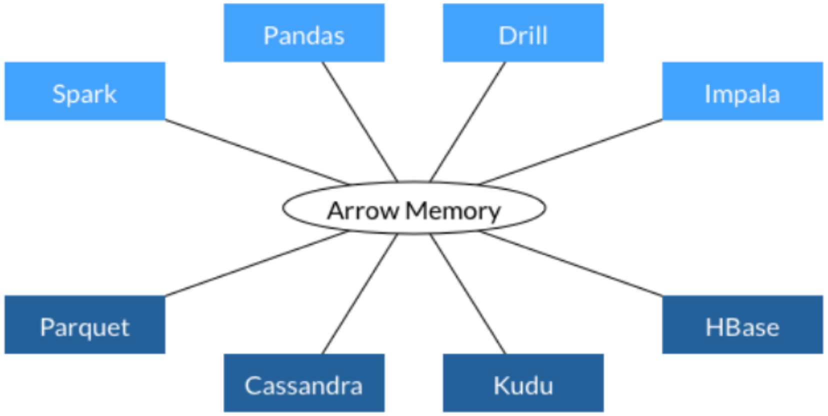
What if we could keep data on the GPU?



LEARNING FROM APACHE ARROW



- Each system has its own internal memory format
- 70-80% computation wasted on serialization and deserialization
- Similar functionality implemented in multiple projects

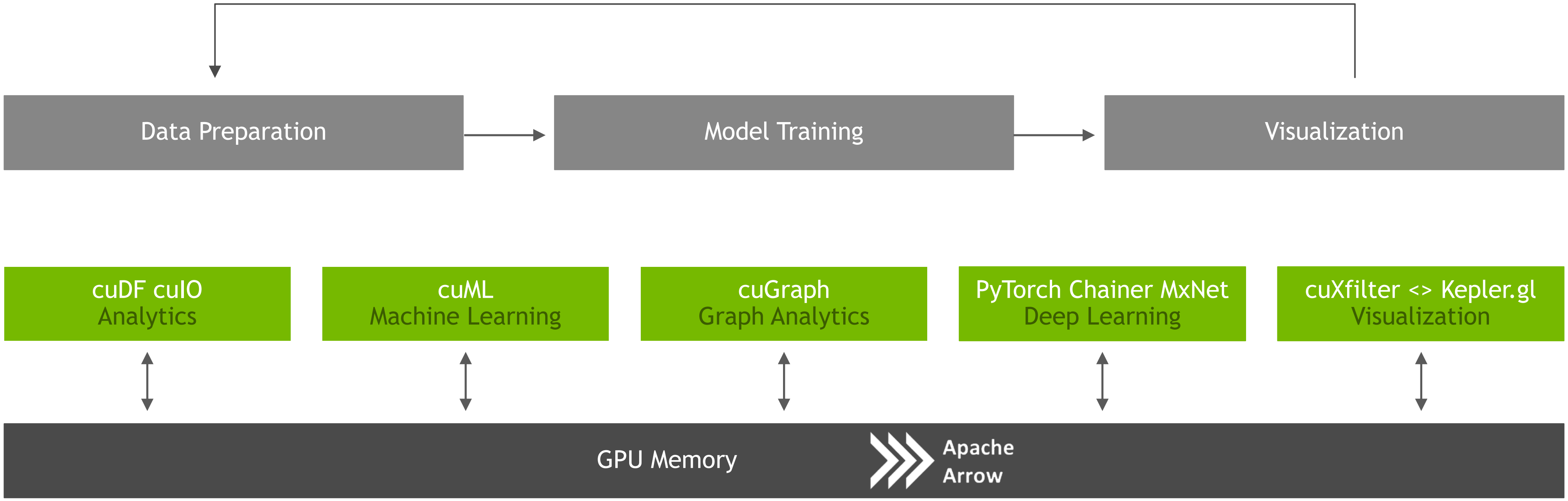


- All systems utilize the same memory format
- No overhead for cross-system communication
- Projects can share functionality (eg, Parquet-to-Arrow reader)

From Apache Arrow Home Page - https://arrow.apache.org/

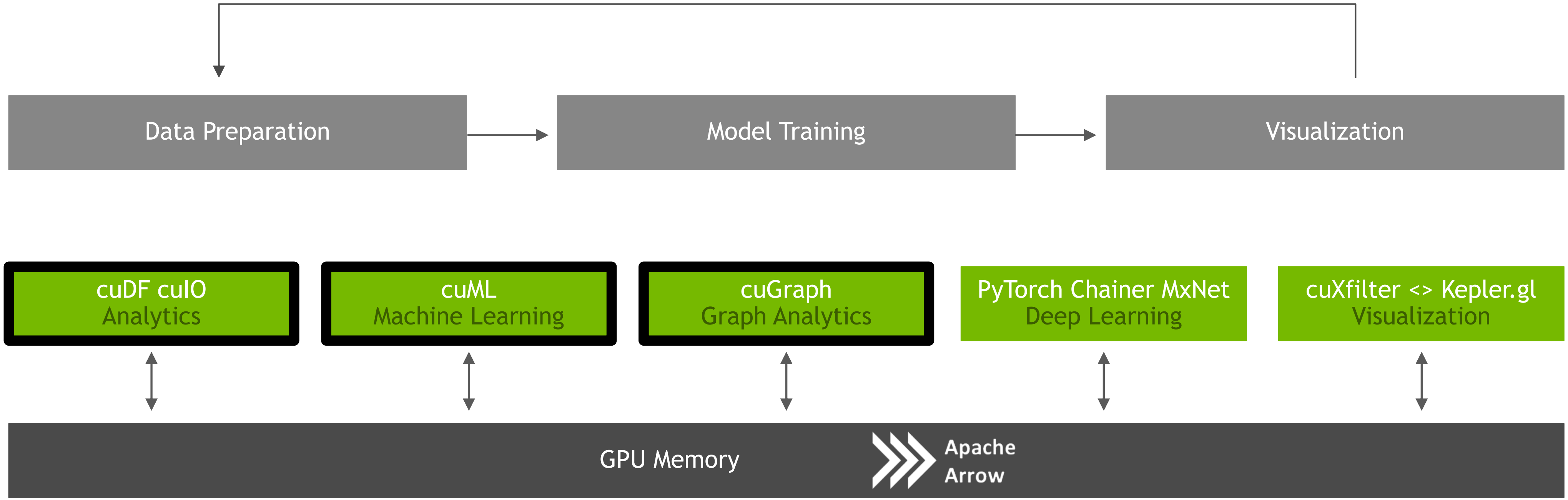
RAPIDS

End to End Accelerate GPU Data Science



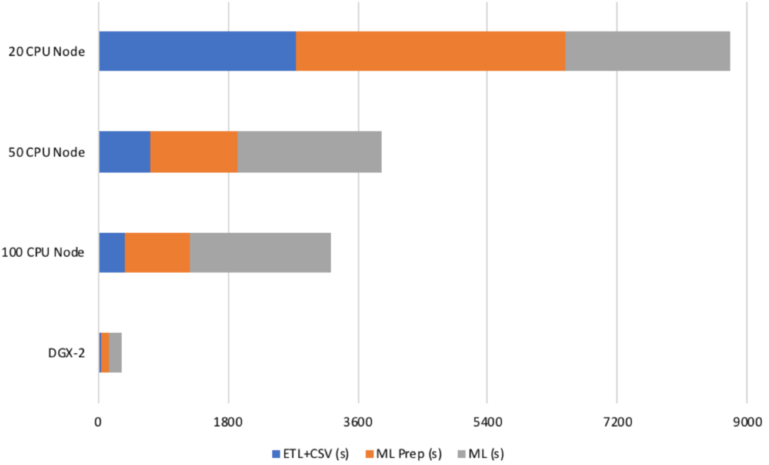
RAPIDS

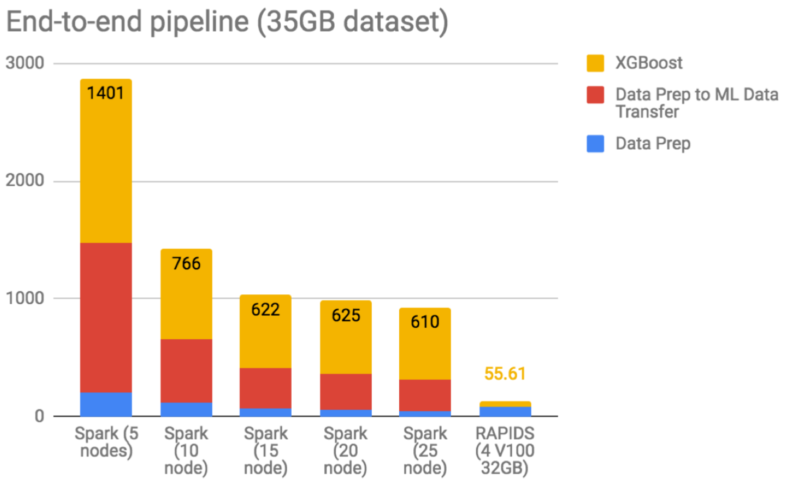
Core of RAPIDS

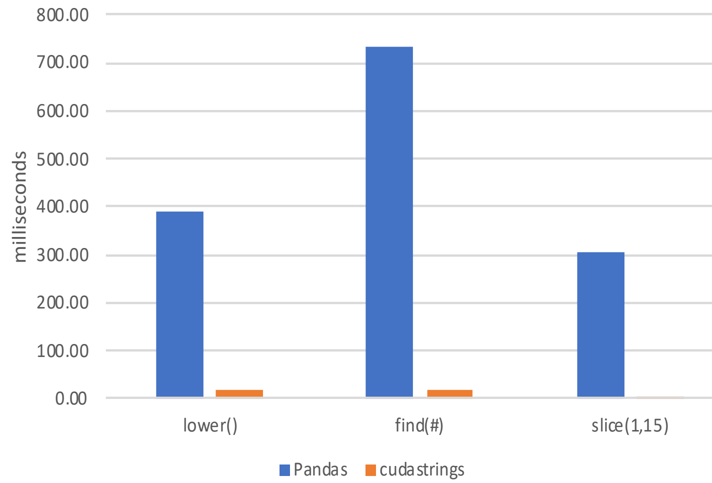


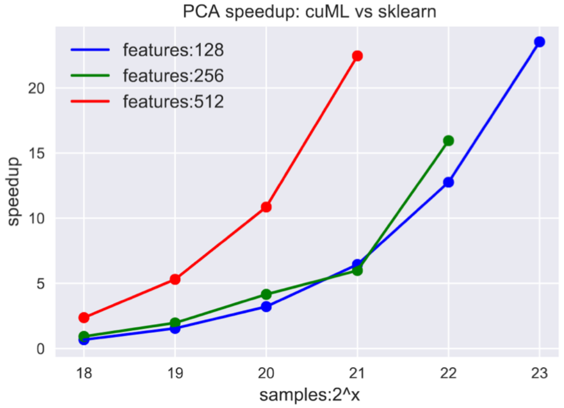
ROAD TO 1.0

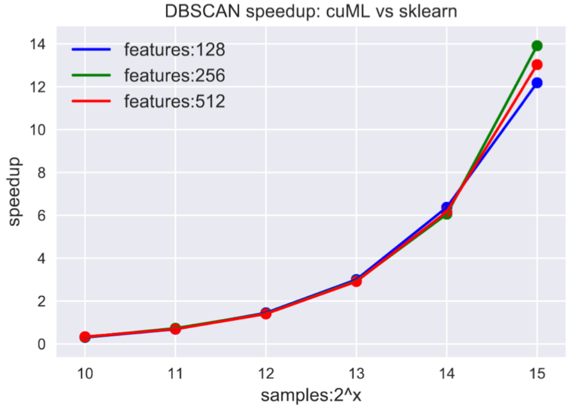
RAPIDS Is Fast…









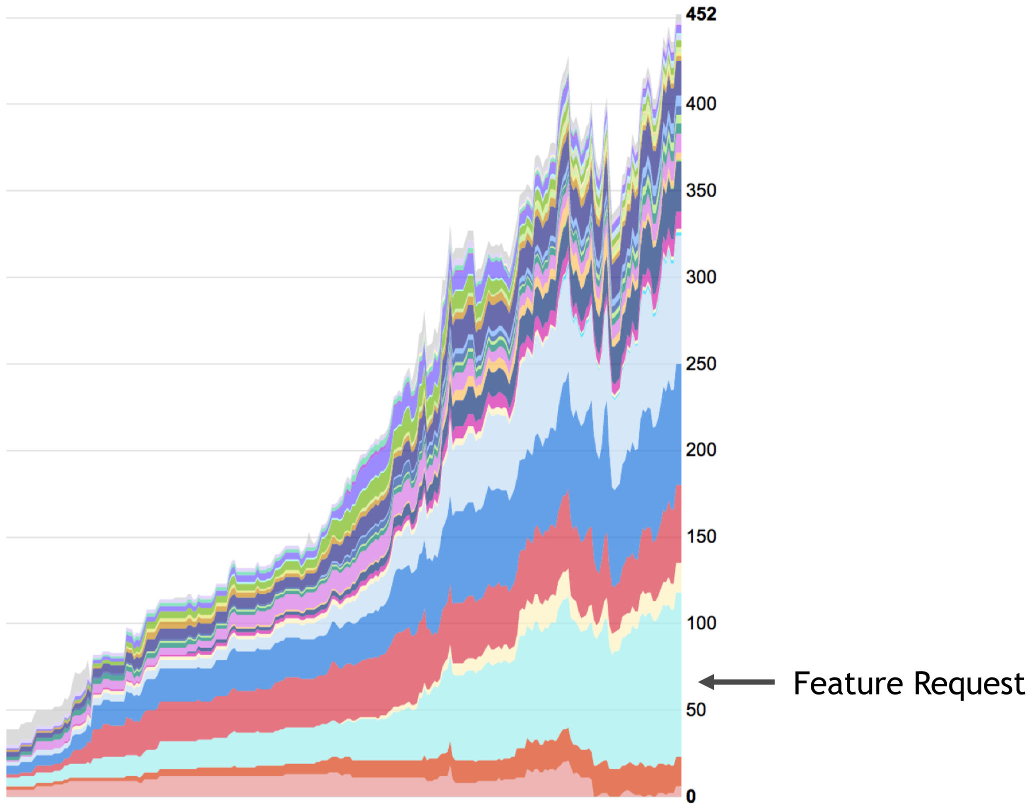


- CPU to GPU 10-25x improvement on average
- Simple Python Interface
- … could be even faster!
  - JIT Compression/Decompression
  - Improved caching
  - More static compiled kernels

NVIDIA.

ROAD TO 1.0

Focused on Robust Functionality, Deployment, and User Experience

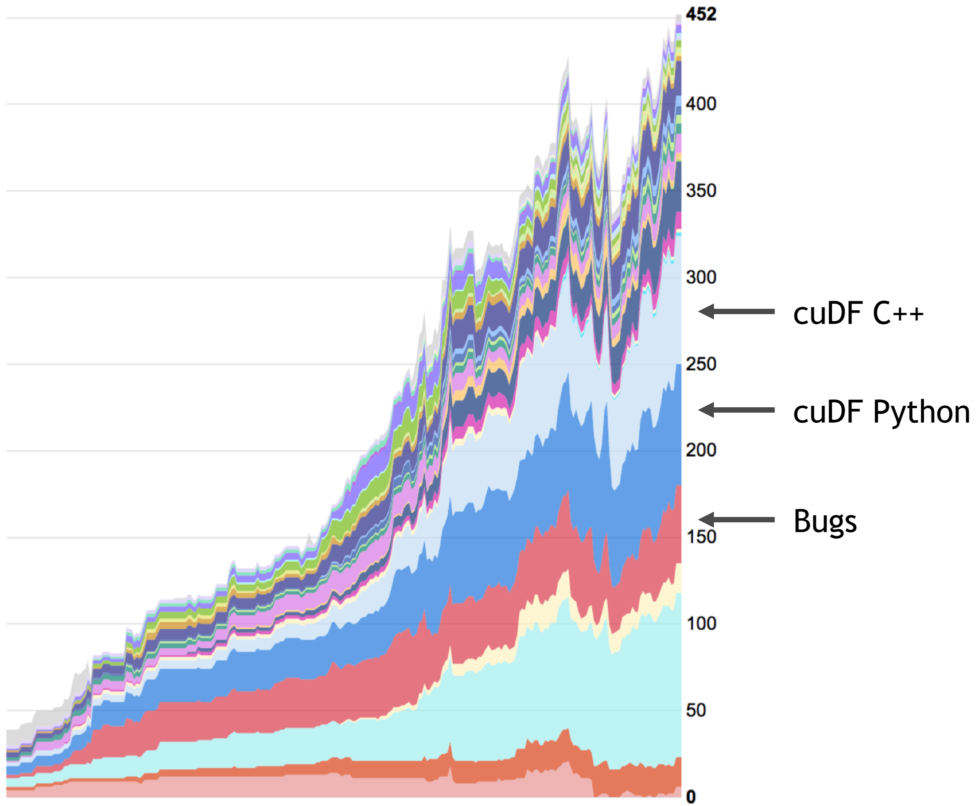


https://9-volt.github.io/bug-life/?repo=rapidsai/cudf

- Window and Rolling Window Functions
- Improved Apply function
- Improved String Support
- Words to Vec
- Geospatial Functions
- Improved Integration with Numba
- Statistical functions (ANOVA, Covariance, etc…)

ROAD TO 1.0

Focused on Robust Functionality, Deployment, and User Experience



https://9-volt.github.io/bug-life/?repo=rapidsai/cudf

- Better error handling
  - Replace CFFI with Cython for more descriptive errors and exceptions
- Cover more edge cases of functionality
- Push more functionality down from cuDF Python into cuDF C++ for performance and future languages
- Support a proper C++ API

ROAD TO 1.0

GTC Europe – Launch - RAPIDS 0.1

cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			

cuGraph	SG	MG	MGMN
Jaccard			
Weighted Jaccard			
PageRank			
Louvain			
SSSP			
BFS			
SSWP			
Triangle Counting			
Subgraph Extraction			

ROAD TO 1.0

GTC San Jose – Today - RAPIDS 0.6

cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			

cuGraph	SG	MG	MGMN
Jaccard			
Weighted Jaccard			
PageRank			
Louvain			
SSSP			
BFS			
SSWP			
Triangle Counting			
Subgraph Extraction			

ROAD TO 1.0

Q4 – 2019 - RAPIDS 0.12?

cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			

cuGraph	SG	MG	MGMN
Jaccard			
Weighted Jaccard			
PageRank			
Louvain			
SSSP			
BFS			
SSWP			
Triangle Counting			
Subgraph Extraction			

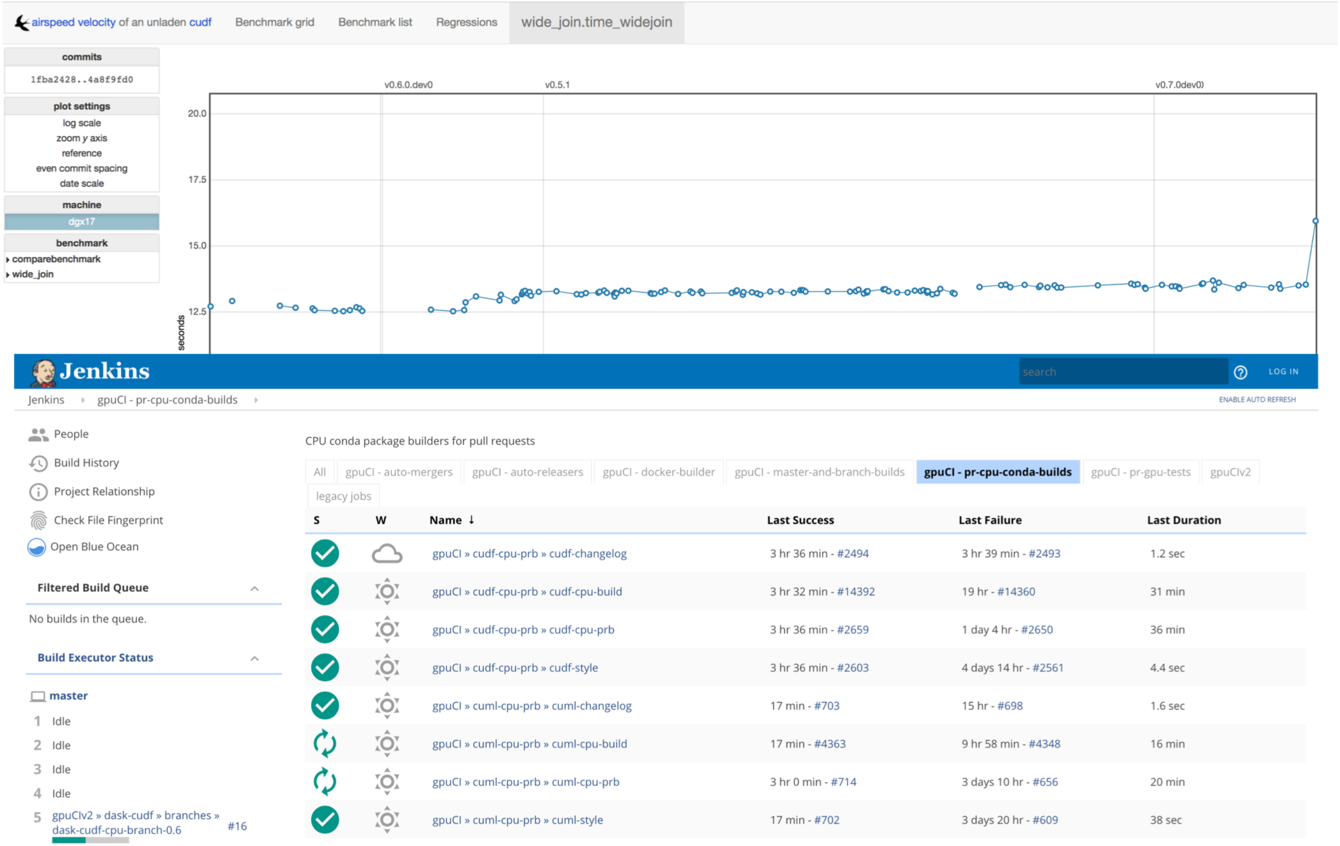
ROAD TO 1.0

Focused on Robust Functionality, Deployment, and User Experience

Integration with every major cloud provider
Both containers and cloud specific machine instances
Support for Enterprise and HPC Orchestration Layers

ROAD TO 1.0

Focused on Robust Functionality, Deployment, and User Experience



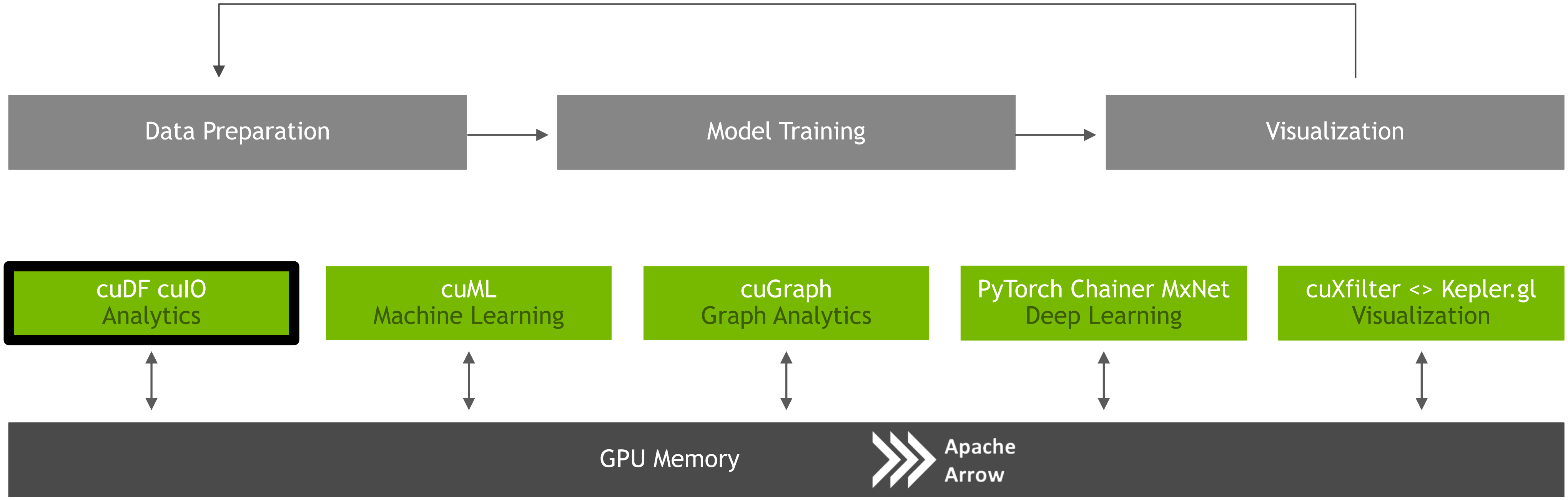
3/19/19 S9788
Building a GPU-Focused CI Solution
Michael Wendt

- CI/CD essential to RAPIDS
- Airspeed Velocity (ASV) for regression
- Nightlies!
  - https://hub.docker.com/r/rapidsai/rapidsai-nightly
  - https://anaconda.org/rapidsai-nightly

NVIDIA

RAPIDS

Core of RAPIDS



ETL - THE BACKBONE OF DATA SCIENCE

cuDF is…

CUDA

- Low level library containing function implementations and C/C++ API
- Importing/exporting Apache Arrow using the CUDA IPC mechanism
- CUDA kernels to perform element-wise math operations on GPU DataFrame columns
- CUDA sort, join, groupby, and reduction operations on GPU DataFrames

	Area Abbreviation	Area Code	Area	Item Code	Item	Element Code	Element	Unit	latitude	longitude	...	Y2004	Y2005	Y2006	Y2007	Y2008
0	AF	2	Afghanistan	2511	Wheat and products	5142	Food	1000 tonnes	33.94	67.71	...	3249.0	3486.0	3704.0	4164.0	4252.0
1	AF	2	Afghanistan	2805	Rice (Milled Equivalent)	5142	Food	1000 tonnes	33.94	67.71	...	419.0	445.0	546.0	455.0	490.0
2	AF	2	Afghanistan	2513	Barley and products	5521	Feed	1000 tonnes	33.94	67.71	...	58.0	236.0	262.0	263.0	230.0
3	AF	2	Afghanistan	2513	Barley and products	5142	Food	1000 tonnes	33.94	67.71	...	185.0	43.0	44.0	48.0	62.0
4	AF	2	Afghanistan	2514	Maize and products	5521	Feed	1000 tonnes	33.94	67.71	...	120.0	208.0	233.0	249.0	247.0
5	AF	2	Afghanistan	2514	Maize and products	5142	Food	1000 tonnes	33.94	67.71	...	231.0	67.0	82.0	67.0	69.0
6	AF	2	Afghanistan	2517	Millet and products	5142	Food	1000 tonnes	33.94	67.71	...	15.0	21.0	11.0	19.0	21.0
7	AF	2	Afghanistan	2520	Cereals, Other	5142	Food	1000 tonnes	33.94	67.71	...	2.0	1.0	1.0	0.0	0.0
8	AF	2	Afghanistan	2531	Potatoes and products	5142	Food	1000 tonnes	33.94	67.71	...	276.0	294.0	294.0	260.0	242.0
9	AF	2	Afghanistan	2536	Sugar cane	5521	Feed	1000 tonnes	33.94	67.71	...	50.0	29.0	61.0	65.0	54.0
10	AF	2	Afghanistan	2537	Sugar beet	5521	Feed	1000 tonnes	33.94	67.71	...	0.0	0.0	0.0	0.0	0.0

With Python Bindings

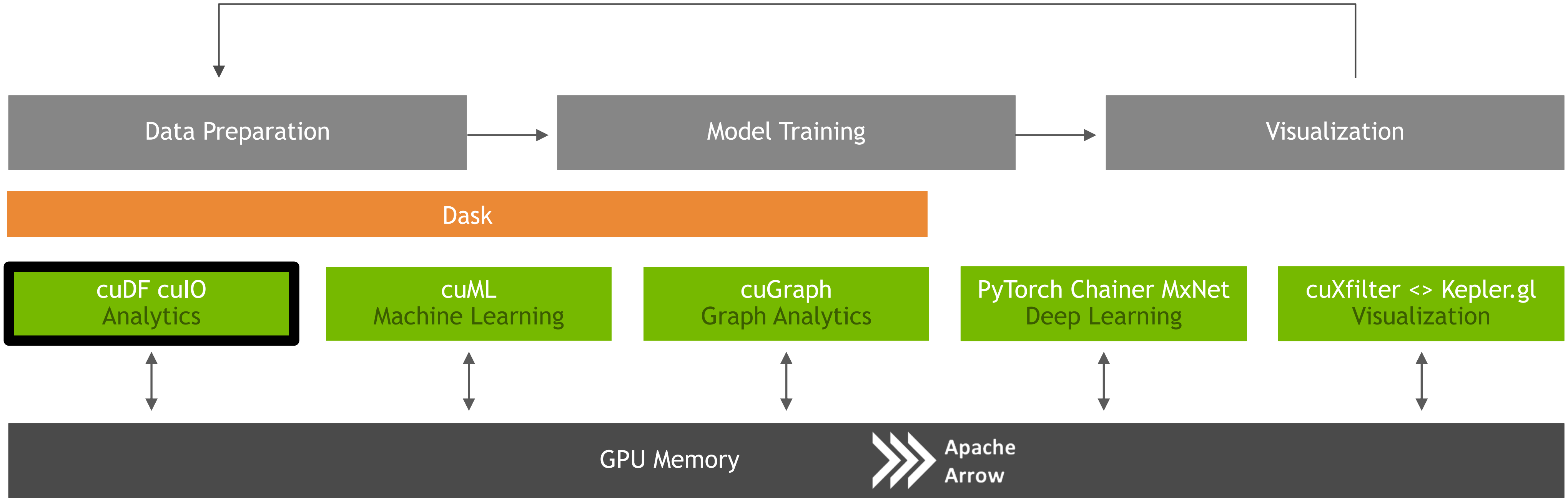
- A Python library for manipulating GPU DataFrames
- Python interface to CUDA C++ with additional functionality
- Creating Apache Arrow from Numpy arrays, Pandas DataFrames, and PyArrow Tables
- JIT compilation of User-Defined Functions (UDFs) using Numba

- Apache Arrow data format
- Pandas-like API

- Unary and Binary Operations
- Joins / Merges
- GroupBys
- Filters
- User-Defined Functions (UDFs)
- Accelerated file readers
- Etc.

ETL - THE BACKBONE OF DATA SCIENCE

cuDF is not the end of the story



3/19/19 S9793 cuDF: RAPIDS GPU-Accelerated Data Frame Library Keith Kraus & Dante Gama Dessavre

3/20/19 RAPIDS CUDA DataFrame Internals for C++ Developers Jake Hemstad

ETL - THE BACKBONE OF DATA SCIENCE

Why Dask

DASK

- **PyData Native**
  - Built on top of NumPy, Pandas Scikit-Learn, … (easy to migrate)
  - With the same APIs (easy to train)
  - With the same developer community (well trusted)
- **Scales**
  - Easy to install and use on a laptop
  - Scales out to thousand-node clusters
- **Popular**
  - Most common parallelism framework today at PyData and SciPy conferences
- **Deployable**
  - HPC: SLURM, PBS, LSF, SGE
  - Cloud: Kubernetes
  - Hadoop/Spark: Yarn

ETL - THE BACKBONE OF DATA SCIENCE

Why Dask

DASK

- **PyData Native**
  - Built on top of NumPy, Pandas Scikit-Learn, … (easy to migrate)
  - With the same APIs (easy to train)
  - With the same developer community (well trusted)
- **Scales**
  - Easy to install and use on a laptop
  - Scales out to thousand-node clusters
- **Popular**
  - Most common parallelism framework today at PyData and SciPy conferences
- **Deployable**
  - HPC: SLURM, PBS, LSF, SGE
  - Cloud: Kubernetes
  - Hadoop/Spark: Yarn

Scott Collis
@Cyclogenesis_au
Follow

Finally got around to learning @dask_dev jobqueue today. Went down to basement, ssh-ed into 10,000 cpu cluster, 10 lines of code and 3 minutes I had a scalable system hooked onto @ProjectJupyter and not a single qsub .. @mrocklin I owe you a beer 🍺

8:22 PM - 27 Aug 2018 from Clarendon Hills, IL

7 Retweets 53 Likes

3 7 53

ETL - THE BACKBONE OF DATA SCIENCE

Dask-cuDF improvements in 0.7 & 0.8

- Make cuDF more Pandas like
  - The more cuDF follows the Pandas API, the fewer changes to Dask DataFrame
- Replace Dask communications with Open UCX
  - Pickling CUDA IPC was a clever hack, but would not scale past a single node
- Focus on Dask-cuDF errors
  - Dask will prevent most out of memory errors users currently experience with cuDF alone
  - Improvements to Dask-cuDF still improve cuDF
- Better memory monitoring in Dask
- Improve String Support…

ETL - THE BACKBONE OF DATA SCIENCE

String Support in Dask-cuDF

Now 0.6 String Support:

- Element-wise operations
  - Split, Find, Extract, Cat, Typecasting, etc…
- String GroupBys
- String Joins
- Power Support now possible

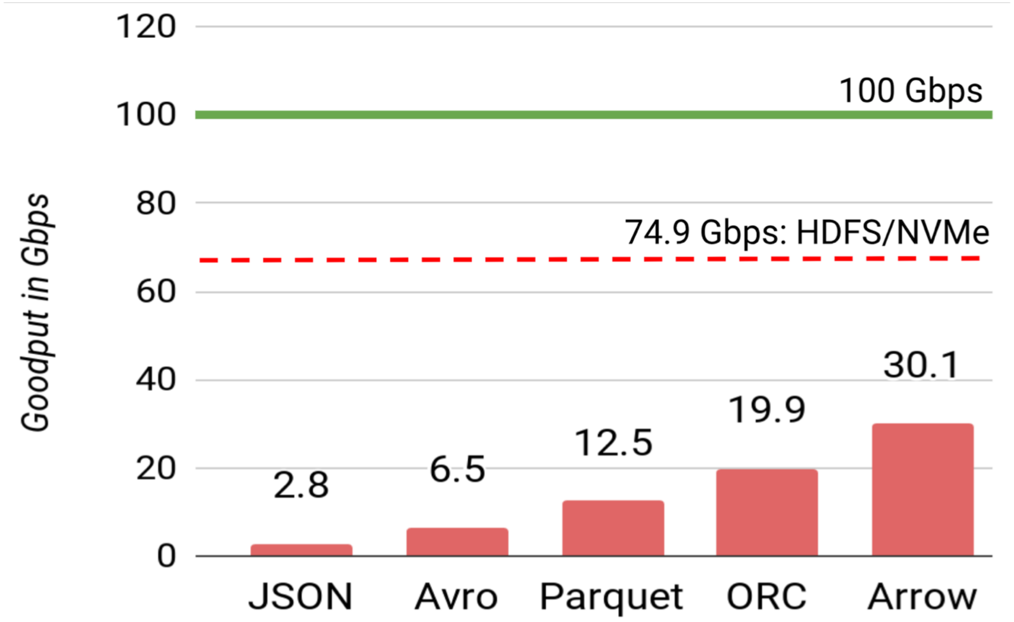
Future 0.7 & 0.8 String Support:

- GPU accelerated to_csv
- More Pandas String API compatibility
- Element-wise String Comparisons
- Improved Categorical column support

EXTRACTION IS THE CORNERSTONE OF ETL

cuIO Is Born

- CSV Reader
  - Follows API of pandas.read_csv
  - Current implementation is >10x speed improvement over pandas
- Parquet Reader – v0.7
  - Work in progress: Will follow API of pandas.read_parquet
- ORC Reader – v0.7
  - Work in progress: Will have similar API of Parquet reader
- Additionally looking towards GPU-accelerating decompression for common compression schemes

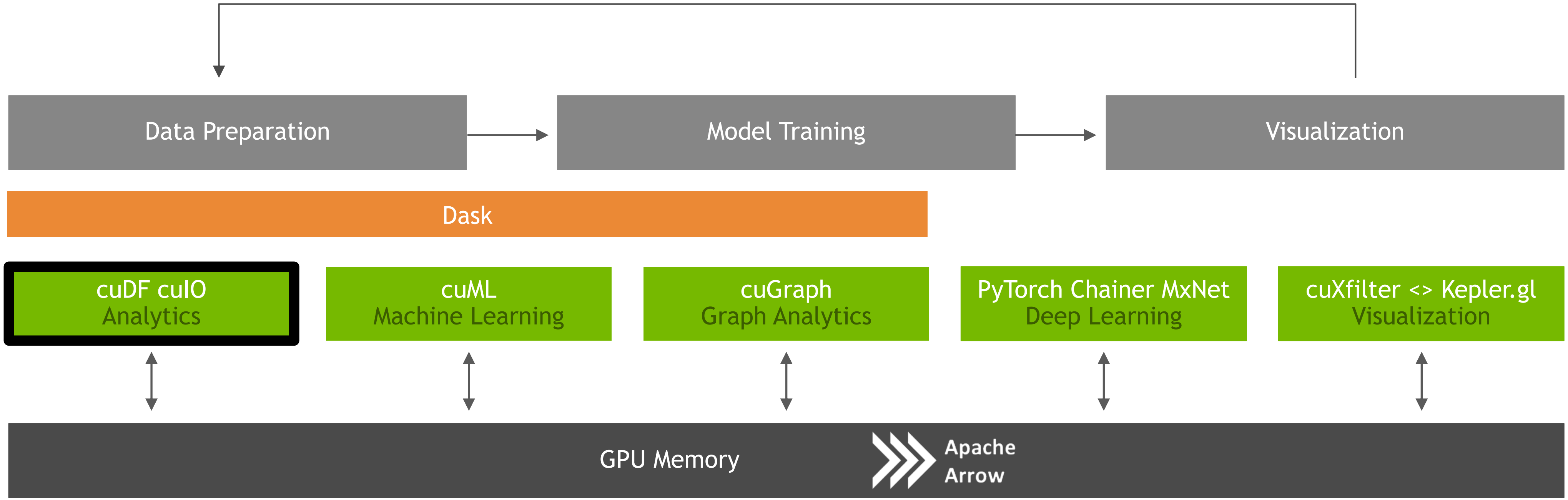


Source: *Apache Crail blog: SQL Performance: Part 1 - Input File Formats*

ETL IS NOT JUST DATAFRAMES!

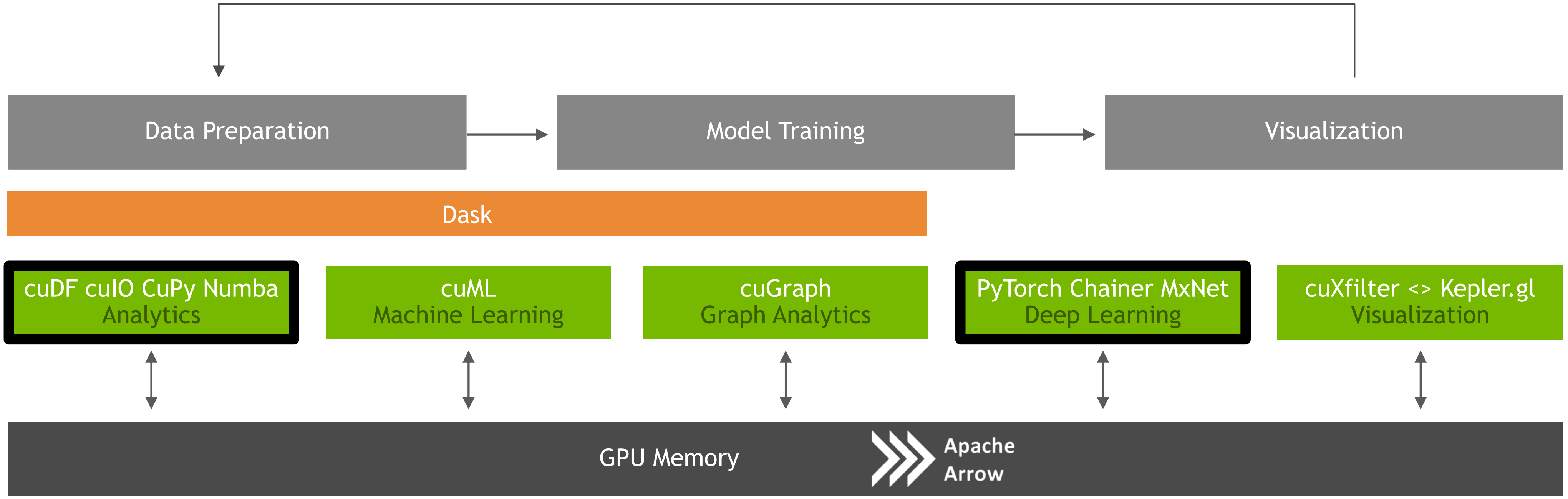
RAPIDS

Core of RAPIDS



RAPIDS

Core of RAPIDS



INTEROPERABILITY FOR THE WIN

DLPack and __cuda_array_interface__

PYTORCH

mxnet

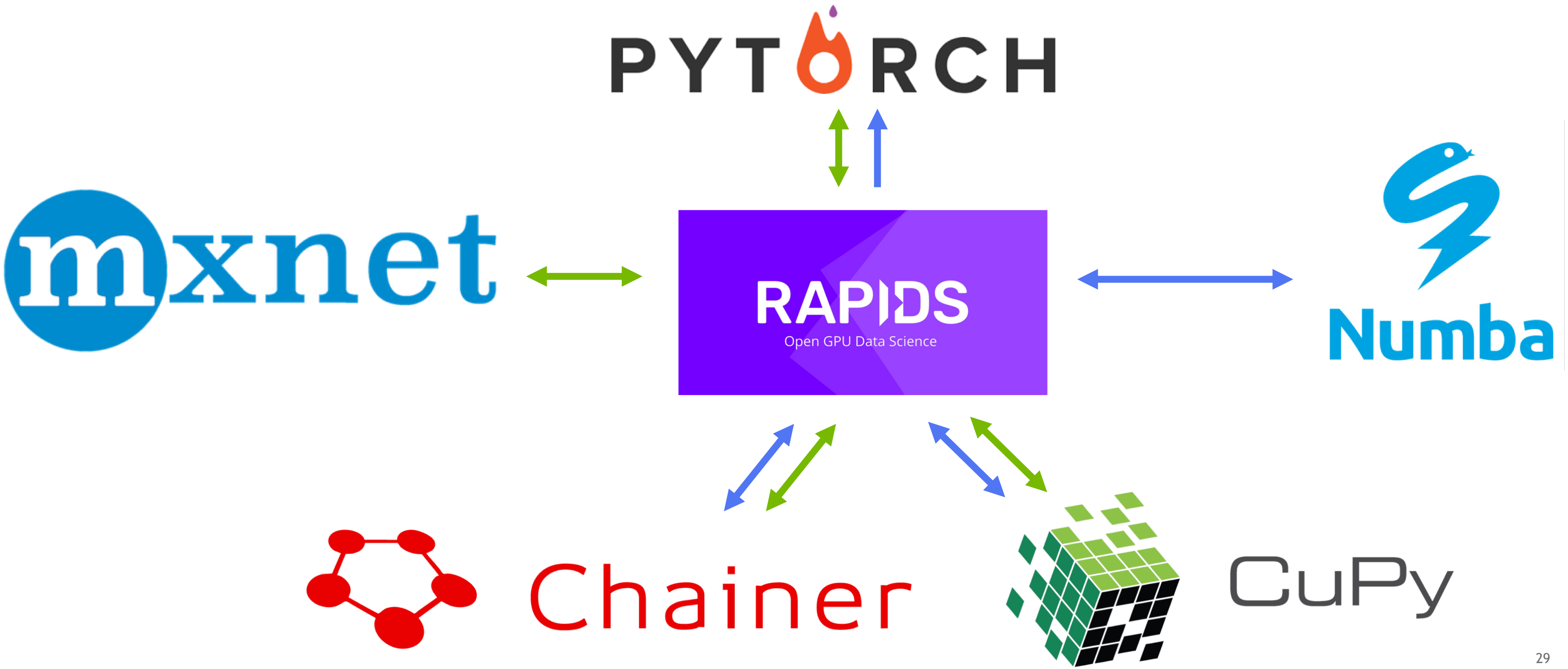
Numba

Chainer

CuPy

INTEROPERABILITY FOR THE WIN

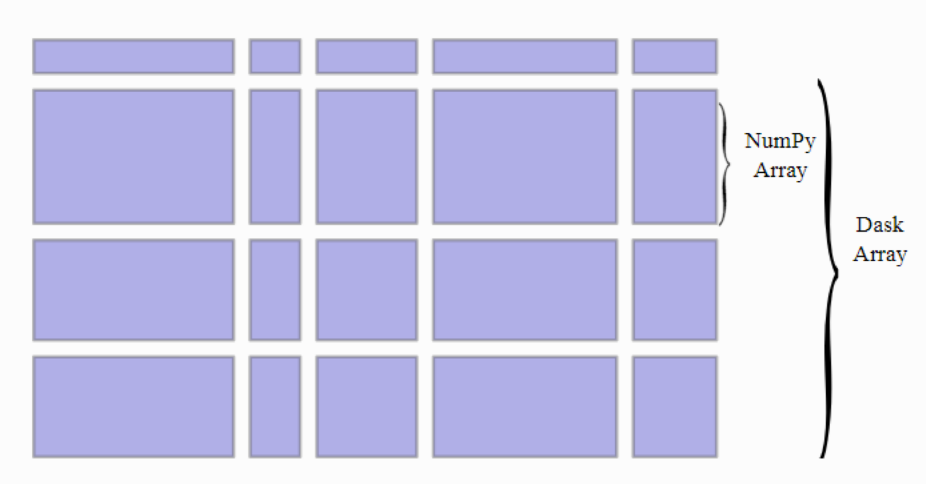
DLPack and __cuda_array_interface__



ETL - ARRAYS & DATAFRAMES

Dask and CUDA Python Arrays

Chainer CuPy Numba



- Scales NumPy to distributed clusters
- Used in climate science, imaging, HPC analysis up to 100TB size
- Now seamlessly accelerated with GPUs

ETL - ARRAYS & DATAFRAMES

Dask-CuPy

Architecture	Time
Single CPU Core	2hr 39min
Forty CPU Cores	11min 30s
One GPU	1 min 37s
Eight GPUs	19s

RAPIDS
Open GPU Data Science

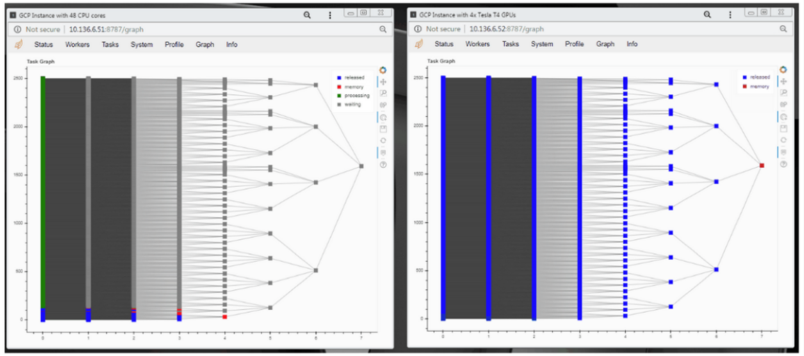
https://blog.dask.org/2019/01/03/dask-array-gpus-first-steps

ETL - ARRAYS & DATAFRAMES

Dask-CuPy

Architecture	Time
Single CPU Core	2hr 39min
Forty CPU Cores	11min 30s
One GPU	1 min 37s
Eight GPUs	19s

https://blog.dask.org/2019/01/03/dask-array-gpus-first-steps



Single node, 48 workers, 2 TB: 11 min 35 s	Single node, 4 workers, 2 TB: 58 seconds
RAPIDS VM Image for GCP **CPU:** 48 vCPU, Hyperthread-enabled Cores per socket: 24 Intel(R) Xeon(R) CPU @ 2.30GHz **Memory:** 384 GB	RAPIDS VM Image for GCP **CPU:** 48 vCPU, Hyperthread-enabled Cores per socket: 24 Intel(R) Xeon(R) CPU @ 2.30GHz **Memory:** 384 GB **GPU:** 4x NVIDIA Tesla T4

https://cloud.google.com/blog/products/ai-machine-learning/nvidias-rapids-joins-our-set-of-deep-learning-vm-images-for-faster-data-science

ETL - ARRAYS & DATAFRAMES

Dask-CuPy

Architecture	Time
Single CPU Core	2hr 39min
Forty CPU Cores	11min 30s
One GPU	1 min 37s
Eight GPUs	19s

https://blog.dask.org/2019/01/03/dask-array-gpus-first-steps

3.2 PETABYTES IN LESS THAN 1 HOUR

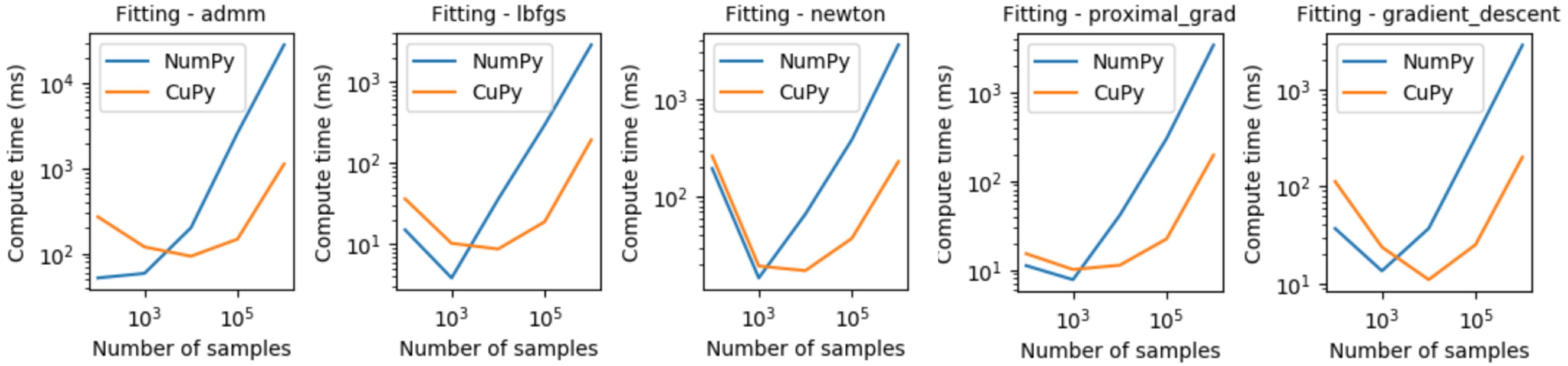
Distributed GPU array | parallel reduction | using 76x GPUs

Array size	Wall Time (data creation + compute)
3.2 PB (20M x 20M doubles)	54 min 51 s

Cluster configuration: 20x GCP instances, each instance has:
CPU: 1 VM socket (Intel Xeon CPU @ 2.30GHz), 2-core, 2 threads/core, 132GB mem, GbE ethernet, 950 GB disk
GPU: 4x NVIDIA Tesla P100-16GB-PCIe (total GPU DRAM across nodes 1.22 TB)
Software: Ubuntu 18.04, RAPIDS 0.5.1, Dask=1.1.1, Dask-Distributed=1.1.1, CuPY=5.2.0, CUDA 10.0.130

ETL - ARRAYS & DATAFRAMES

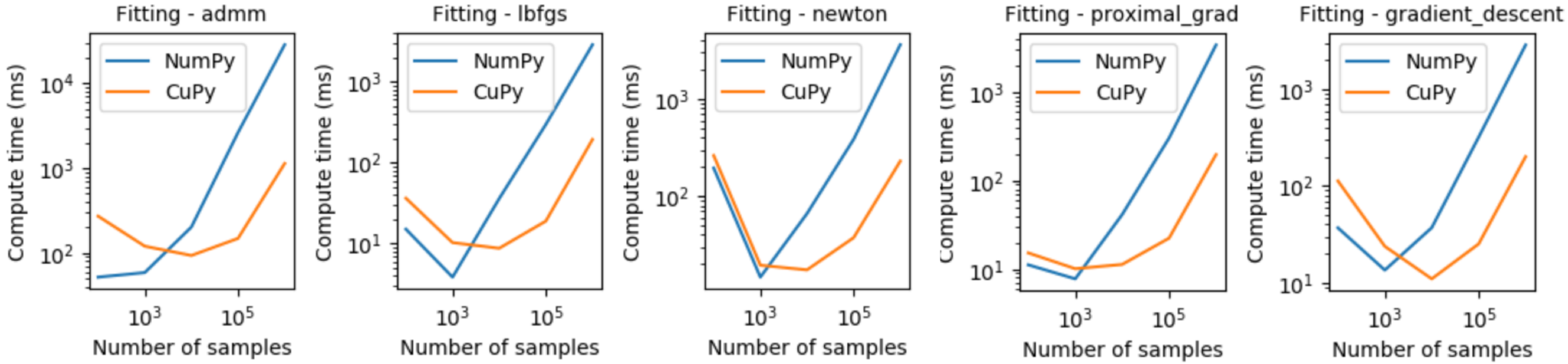
Dask-CuPy




```python
import numpy as np
from dask_glm.estimators import LinearRegression
import matplotlib.pyplot as plt

N = 1000

# x from 0 to N
x = N * np.random.random((40000, 1))

# y = a*x + b with noise
y = 0.5 * x + 1.0 + np.random.normal(size=x.shape)

# create a linear regression model
est = LinearRegression(solver='lbfgs')
```

ETL - ARRAYS & DATAFRAMES

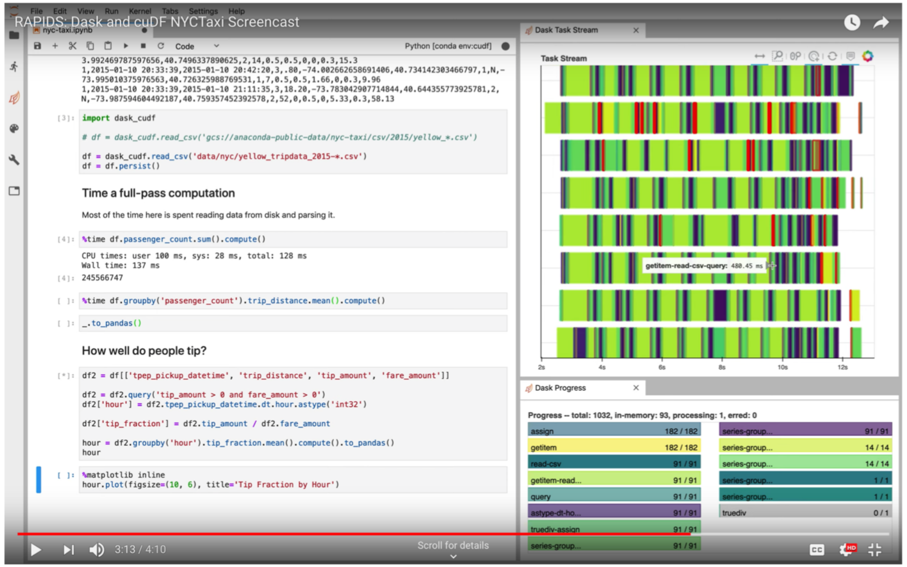
Dask-CuPy is Easy to Implement



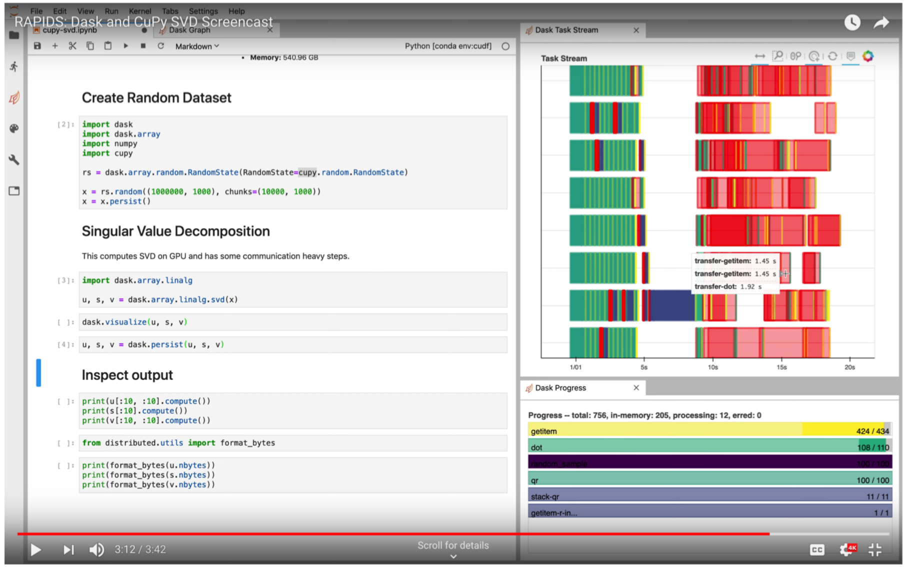

```python
import cupy
from dask_glm.estimators import LinearRegression
import matplotlib.pyplot as plt

N = 1000

# x from 0 to N
x = N * cupy.random.random((40000, 1))

# y = a*x + b with noise
y = 0.5 * x + 1.0 + cupy.random.normal(size=x.shape)

# create a linear regression model
est = LinearRegression(solver='lbfgs')
```

ETL - ARRAYS & DATAFRAMES

More Dask Awesomeness from RAPIDS



https://youtu.be/gV0cykgsTPM

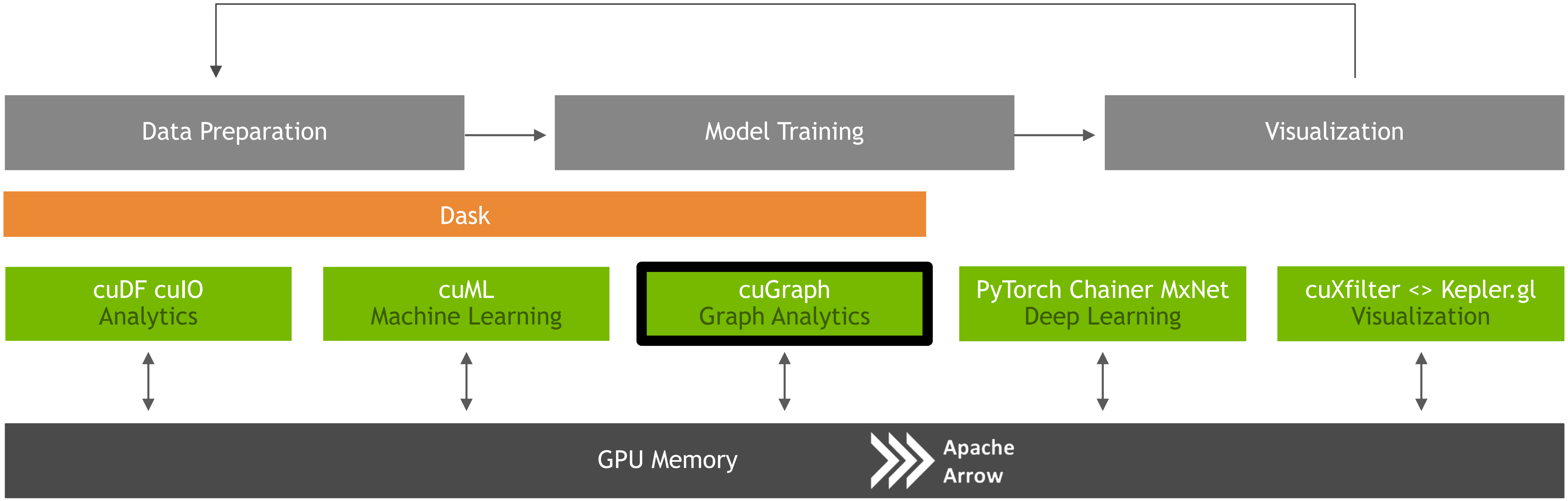


https://youtu.be/R5CiXti_MWo

3/19/19 S9797 Dask Extensions and New Developments with RAPIDS Matthew Rockling

NEW FEATURES WITH GRAPH

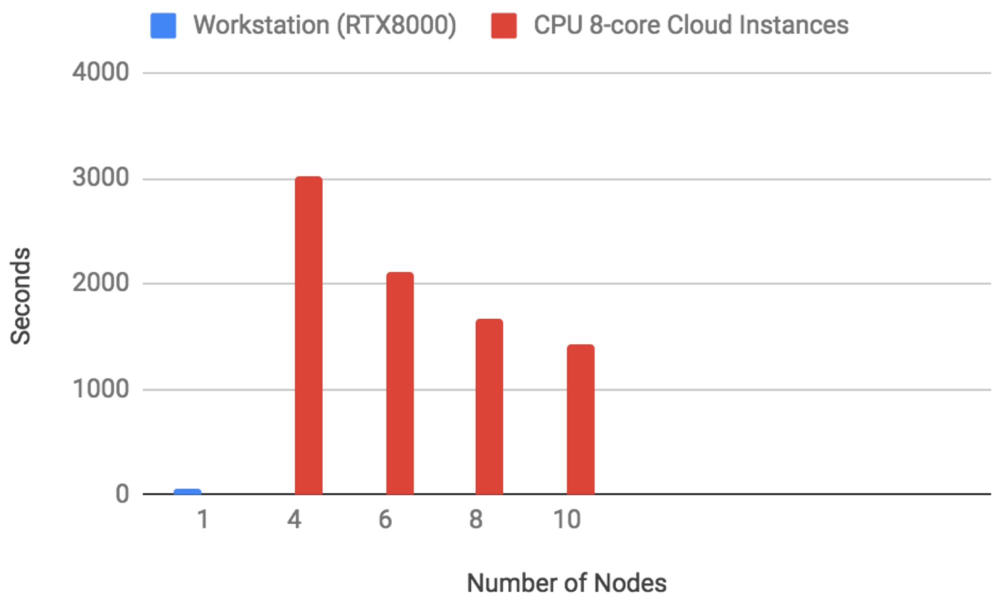
Connected Data the Next Frontier



PAGERANK

1 RTX8000 or Many CPU Nodes

cuGraph	SG	MG	MGMN
Jaccard			
Weighted Jaccard			
PageRank			
Louvain			
SSSP			
BFS			
SSWP			
Triangle Counting			
Subgraph Extraction			



PAGERANK

Future of Graph

cuGraph	SG	MG	MGMN
Jaccard			
Weighted Jaccard			
PageRank			
Louvain			
SSSP			
BFS			
SSWP			
Triangle Counting			
Subgraph Extraction			

- For PageRank Less than 3 seconds spent in algorithm!
- Multi-GPU with Dask
- Better Graph Partitioning on GPU

DASK

PAGERANK

1 RTX8000 or Many CPU Nodes

cuGraph	SG	MG	MGMN
Jaccard			
Weighted Jaccard			
PageRank			
Louvain			
SSSP			
BFS			
SSWP			
Triangle Counting			
Subgraph Extraction			

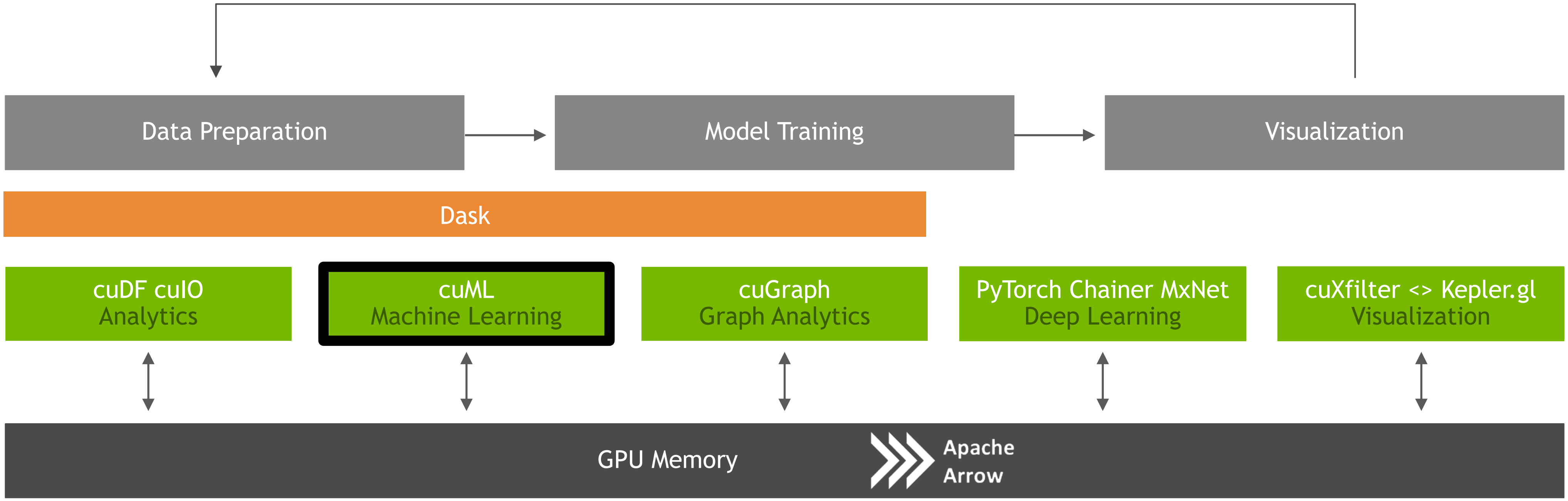
More Talks on Graph!

3/21/19 S9802
Context-Aware Network Mapping and Asset Classification
Bartley Richardson

3/21/19 S9783
Accelerating Graph Algorithms with RAPIDS
Joe Eaton & Brad Rees

MACHINE LEARNING

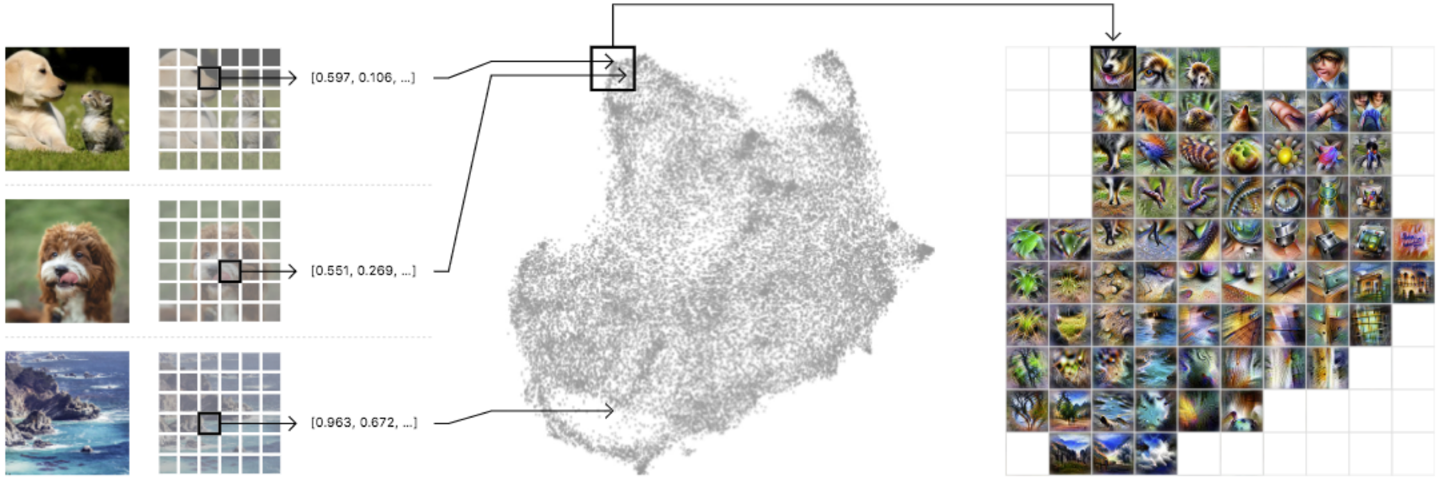
More Models More Problem



UMAP

New Dimensionality Reduction on GPU

Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualisation similarly to t-SNE, but also for general non-linear dimension reduction.



https://ai.googleblog.com/2019/03/exploring-neural-networks.html

Thank You Leland McInnes!

- First of all UMAP is *fast*… scaling beyond what most t-SNE packages can manage…
- Second, UMAP scales well in embedding dimension -- it isn't just for visualisation! You can use UMAP as a general purpose dimension reduction technique…
- Third, UMAP often performs better at preserving aspects of global structure of the data than t-SNE…
- Fourth, UMAP supports a wide variety of distance functions…
- Fifth, UMAP supports adding new points to an existing embedding via the standard sklearn transform method…
- Sixth, UMAP supports supervised and semi-supervised dimension reduction…
- Finally UMAP has solid theoretical foundations in manifold learning…

https://arxiv.org/pdf/1802.03426.pdf

CUML

Single GPU and XGBoost

cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			

DASK-CUML

OLS, tSVD, and KNN in RAPIDS 0.6

cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			

DASK-CUML

K-Means*, DBSCAN & PCA in RAPIDS 0.7/0.8

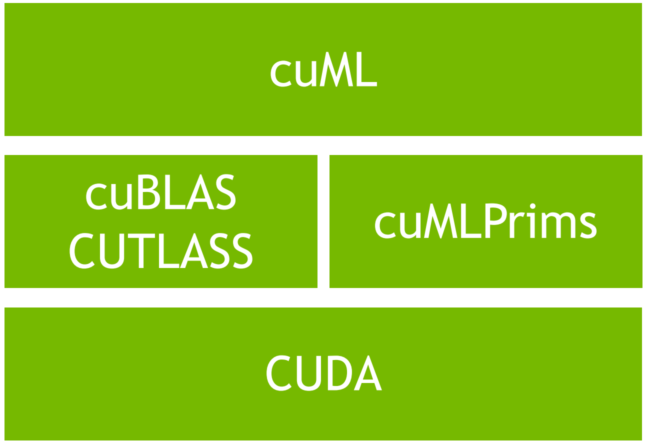
cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			

- Deprecating the current K-means in 0.6 for new K-means built on MLPrims

DASK-CUML

cuML and cuMLPrims

cuML	SG	MG	MGMN
Gradient Boosted Decision Trees (GBDT)			
GLM			
Logistic Regression			
Random Forest (regression)			
K-Means			
K-NN			
DBSCAN			
UMAP			
ARIMA			
Kalman Filter			
Holts-Winters			
Principal Components			
Singular Value Decomposition			



3/20/19 S9817
RAPIDS cuML: GPU Accelerated Machine Learning
Onur Yilmaz & Corey Nolet

RAPIDS

How do I get the software?

- https://github.com/rapidsai
- https://anaconda.org/rapidsai/
- https://pypi.org/project/cudf
- https://pypi.org/project/cuml

- https://ngc.nvidia.com/registry/nvidia-rapidsai-rapidsai
- https://hub.docker.com/r/rapidsai/rapidsai/

JOIN THE MOVEMENT

Everyone Can Help!

APACHE ARROW	RAPIDS	GPU Open Analytics Initiative
https://arrow.apache.org/	https://rapids.ai	http://gpuopenanalytics.com/
@ApacheArrow	**@RAPIDSAI**	**@GPUOAI**

Integrations, feedback, documentation support, pull requests, new issues, or code donations welcomed!

THANK YOU

Joshua Patterson @datametrician