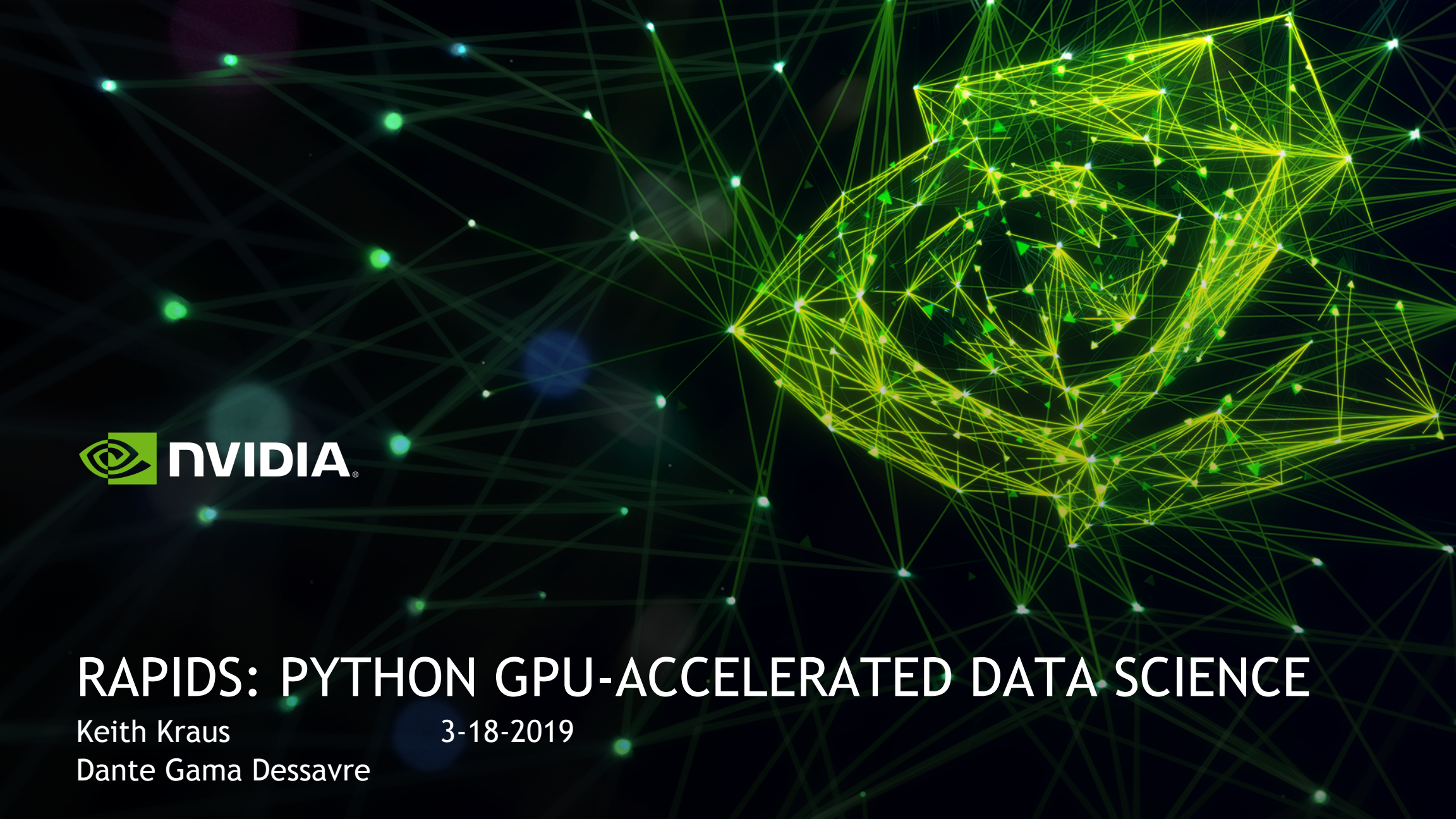


RAPIDS: PYTHON GPU-ACCELERATED DATA SCIENCE

Keith Kraus
Dante Gama Dessavre

3-18-2019

DATA PROCESSING EVOLUTION

Faster Data Access Less Data Movement

Hadoop Processing, Reading from disk

NVIDIA.

DATA PROCESSING EVOLUTION

Faster Data Access Less Data Movement

Hadoop Processing, Reading from disk

Spark In-Memory Processing

25-100x Improvement
Less code
Language flexible
Primarily In-Memory

WE NEED MORE COMPUTE!

Basic workloads are bottlenecked by the CPU

- In a simple benchmark consisting of aggregating data, the **CPU is the bottleneck**
- This is after the data is parsed and cached into memory which is another common bottleneck
- The CPU bottleneck is even worse in more complex workloads!

SELECT cab_type, count(*) FROM trips_orc GROUP BY cab_type;

```
top - 08:54:14 up  1:50,  4 users,  load average: 0.20, 1.64, 6.43
Tasks: 360 total,   2 running, 358 sleeping,   0 stopped,   0 zombie
%Cpu0  : 94.7 us,  1.7 sy,  0.0 ni,  3.3 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0
%Cpu1  : 95.0 us,  1.7 sy,  0.0 ni,  3.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu2  : 98.3 us,  0.3 sy,  0.0 ni,  1.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu3  : 87.3 us,  4.3 sy,  0.0 ni,  8.4 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu4  : 95.0 us,  1.3 sy,  0.0 ni,  3.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu5  : 98.3 us,  0.0 sy,  0.0 ni,  1.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu6  : 96.7 us,  1.3 sy,  0.0 ni,  2.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu7  : 92.7 us,  1.0 sy,  0.0 ni,  5.6 id,  0.3 wa,  0.0 hi,  0.3 si,  0.0
%Cpu8  : 93.7 us,  1.3 sy,  0.0 ni,  5.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu9  : 92.3 us,  0.7 sy,  0.0 ni,  7.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu10 : 97.3 us,  0.7 sy,  0.0 ni,  2.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu11 : 97.3 us,  0.7 sy,  0.0 ni,  2.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu12 : 92.0 us,  3.0 sy,  0.0 ni,  5.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu13 : 94.9 us,  1.0 sy,  0.0 ni,  4.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu14 : 88.3 us,  3.0 sy,  0.0 ni,  8.7 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu15 : 92.6 us,  2.3 sy,  0.0 ni,  4.7 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0
%Cpu16 : 94.7 us,  2.3 sy,  0.0 ni,  2.6 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0
%Cpu17 : 93.0 us,  0.7 sy,  0.0 ni,  6.0 id,  0.0 wa,  0.0 hi,  0.3 si,  0.0
%Cpu18 : 93.0 us,  3.7 sy,  0.0 ni,  3.3 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
%Cpu19 : 91.2 us,  0.7 sy,  0.0 ni,  8.1 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0
```

Source: Mark Litwintschik's blog: 1.1 Billion Taxi Rides: EC2 versus EMR

NVIDIA.

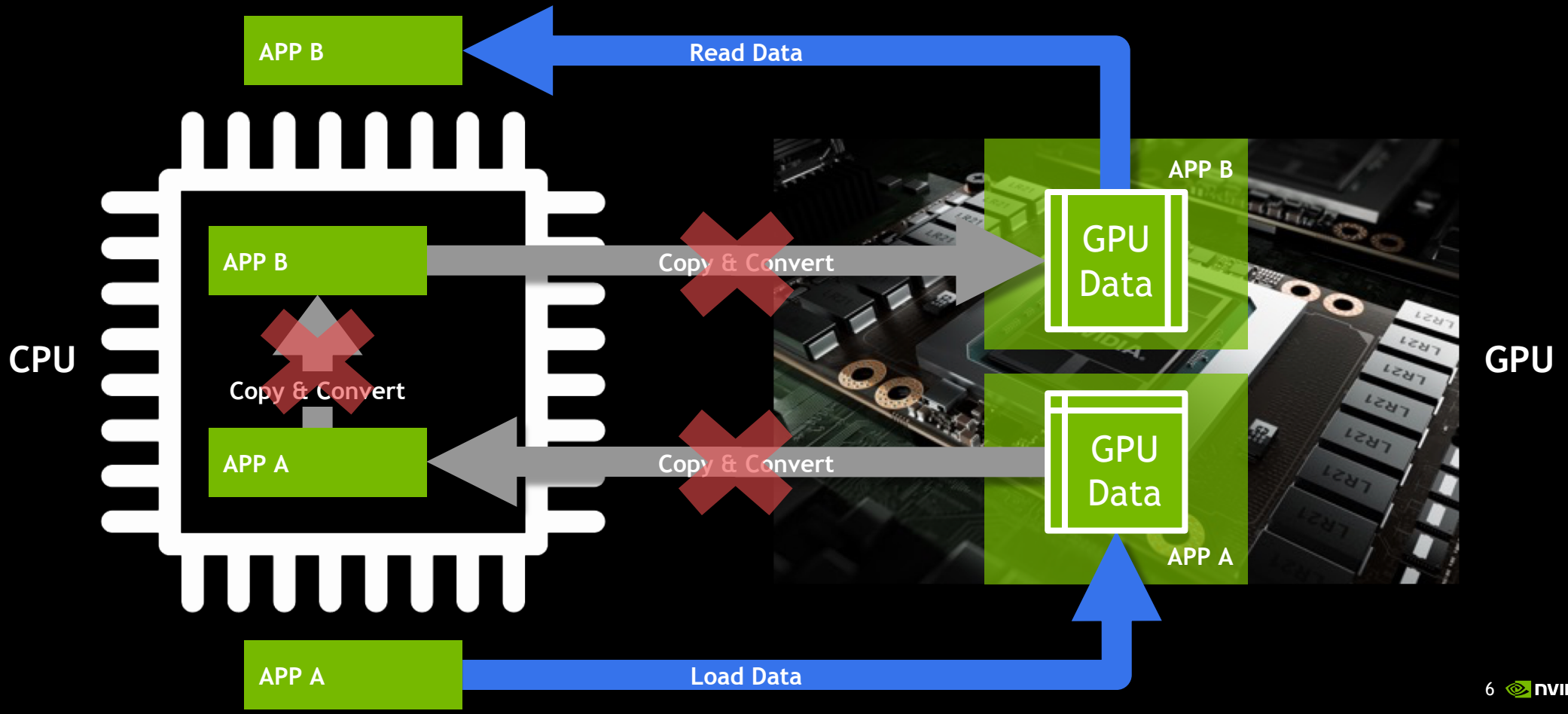
HOW CAN WE DO BETTER?

- Focus on the full Data Science workflow
  - Data Loading
  - Data Transformation
  - Data Analytics

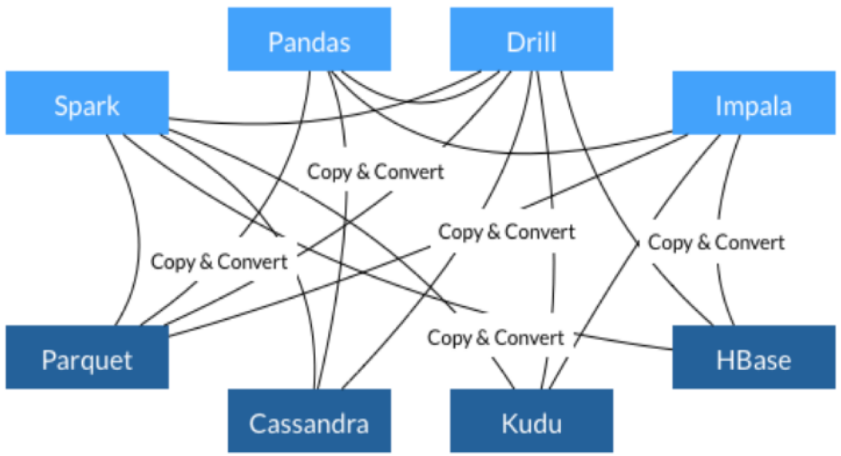
- Python
  - Provide as close to a drop-in replacement for existing tools
- Performance - Leverage GPUs

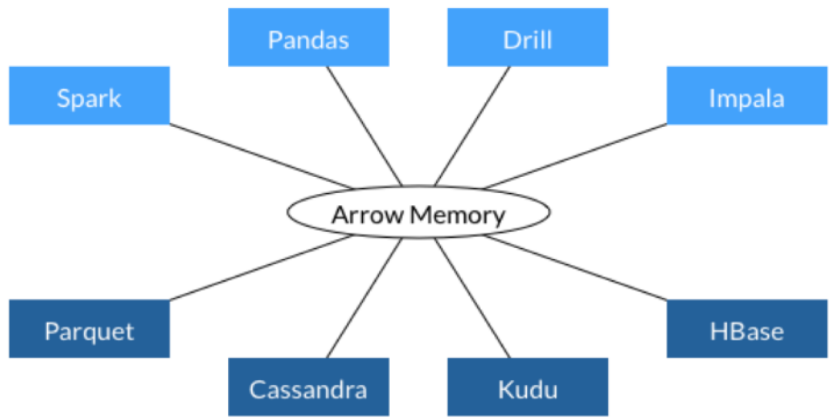
DATA MOVEMENT AND TRANSFORMATION

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

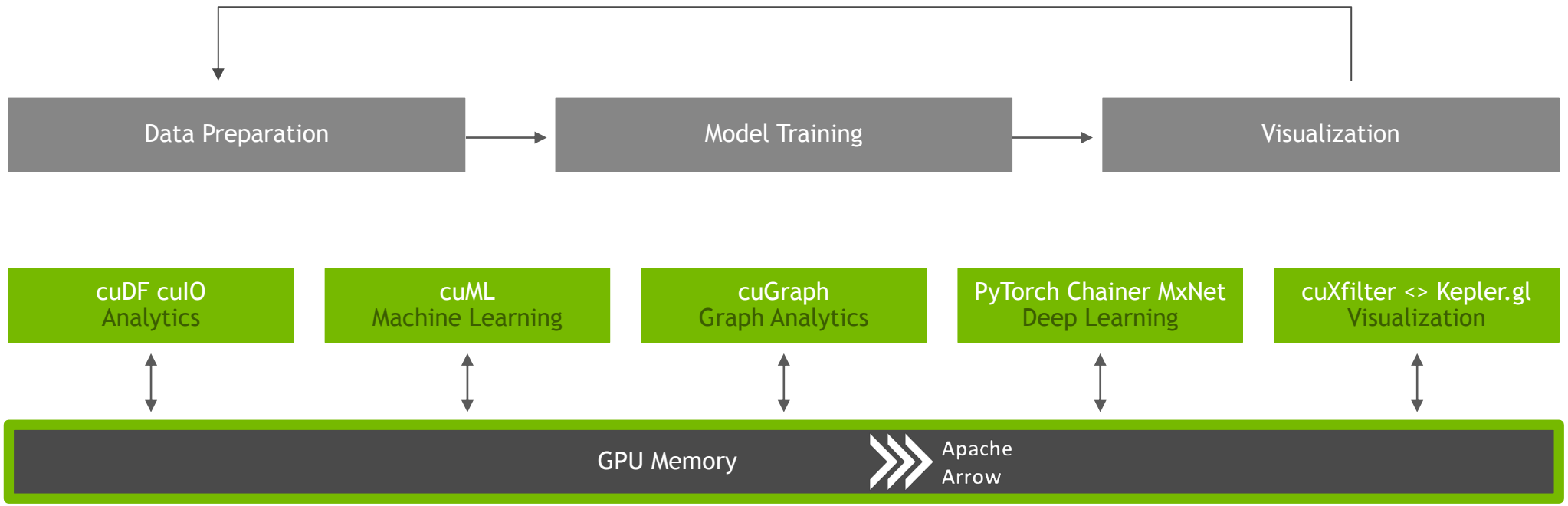
- All systems utilize the same memory format
- No overhead for cross-system communication
- Projects can share functionality (eg, Parquet-to-Arrow reader)

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

NVIDIA

RAPIDS

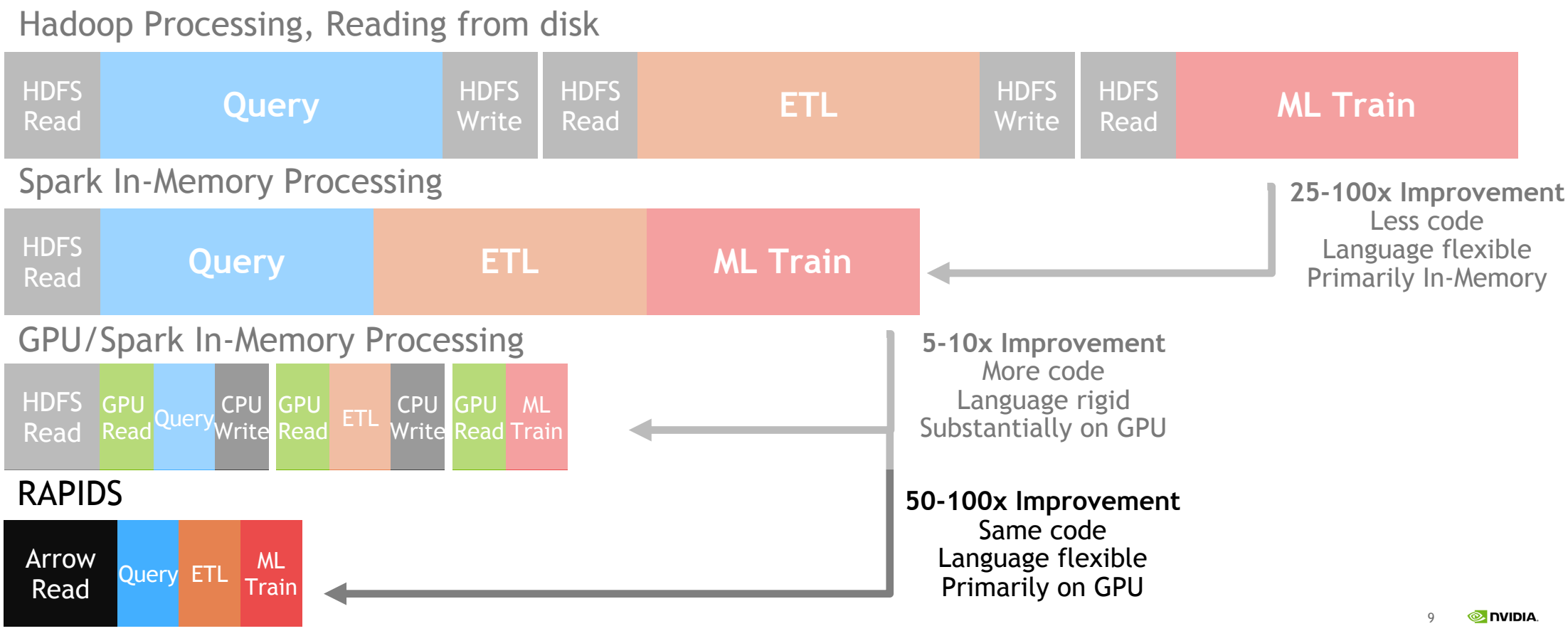
End to End Accelerate GPU Data Science



NVIDIA.

DATA PROCESSING EVOLUTION

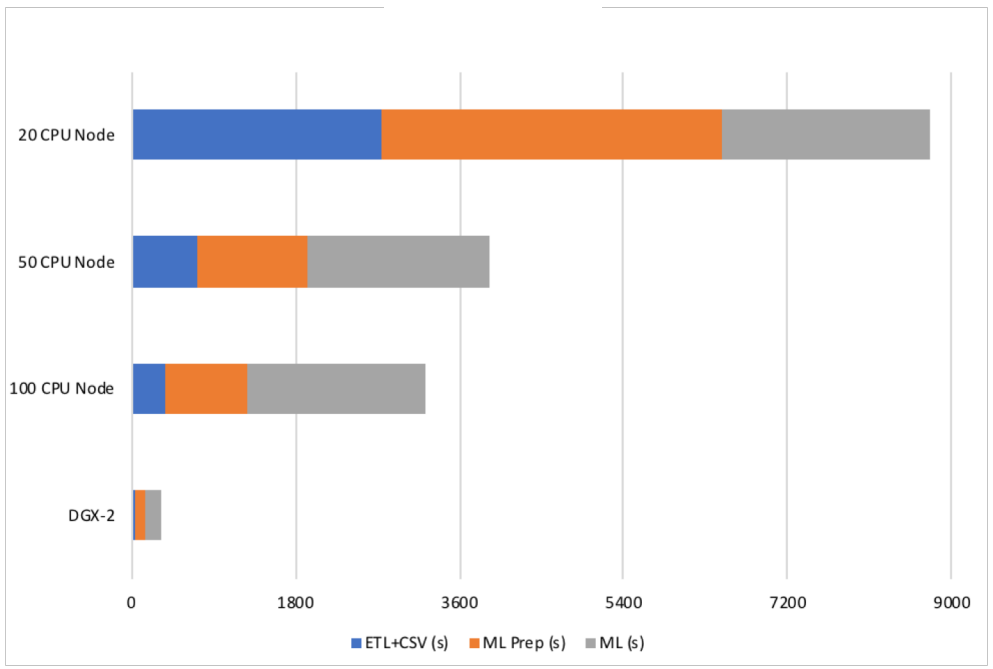
Faster Data Access Less Data Movement

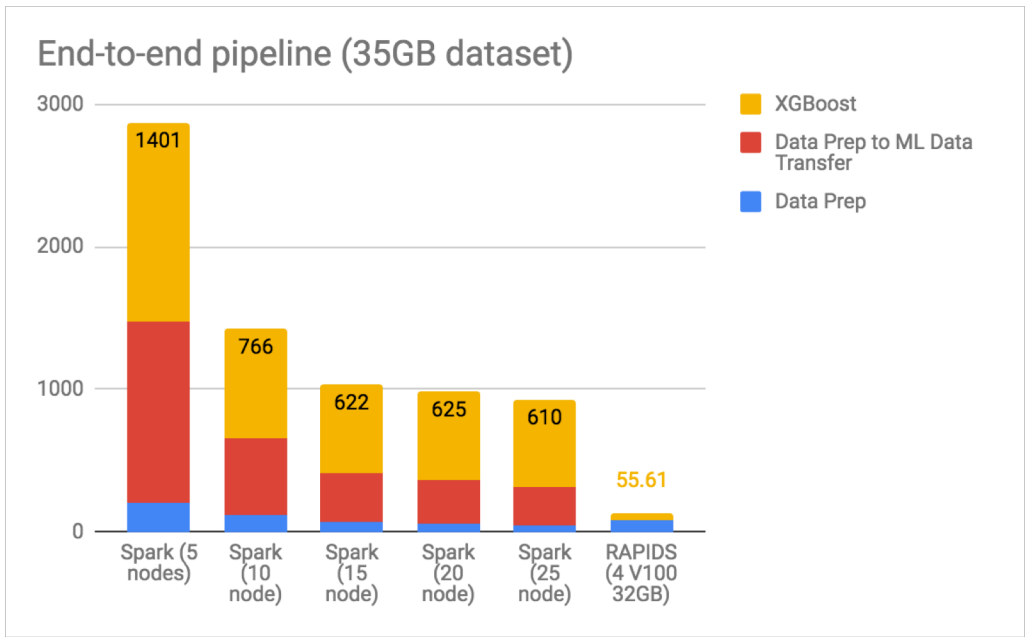


NVIDIA.

THE NEED FOR SPEED

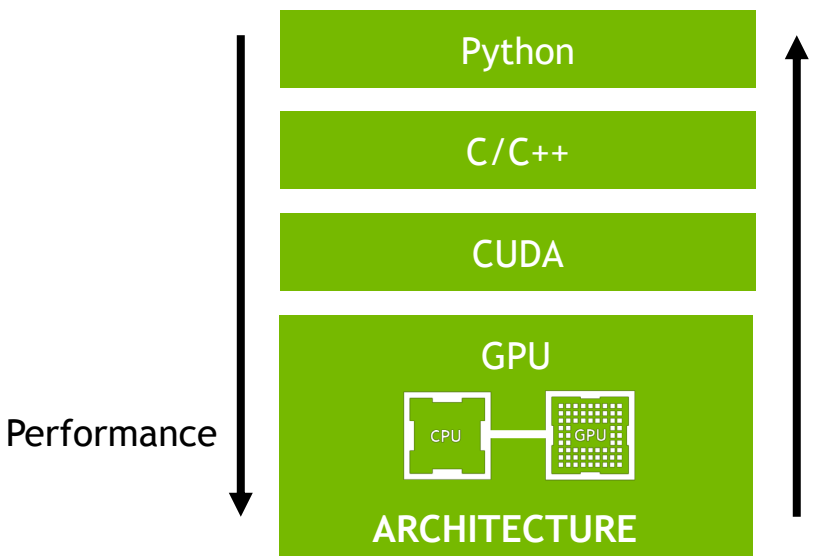
RAPIDS is fast… but could be even faster!





WITHOUT SACRIFICING USABILITY

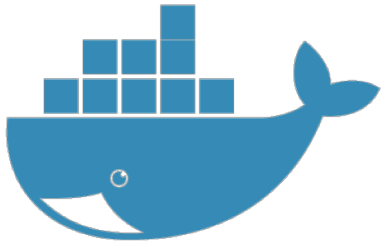
RAPIDS needs to be friendly for every data scientist



- RAPIDS delivers the performance of GPU-Accelerated CUDA
- RAPIDS delivers the ease of use of the Python data science ecosystem

RAPIDS

Install anywhere and everywhere

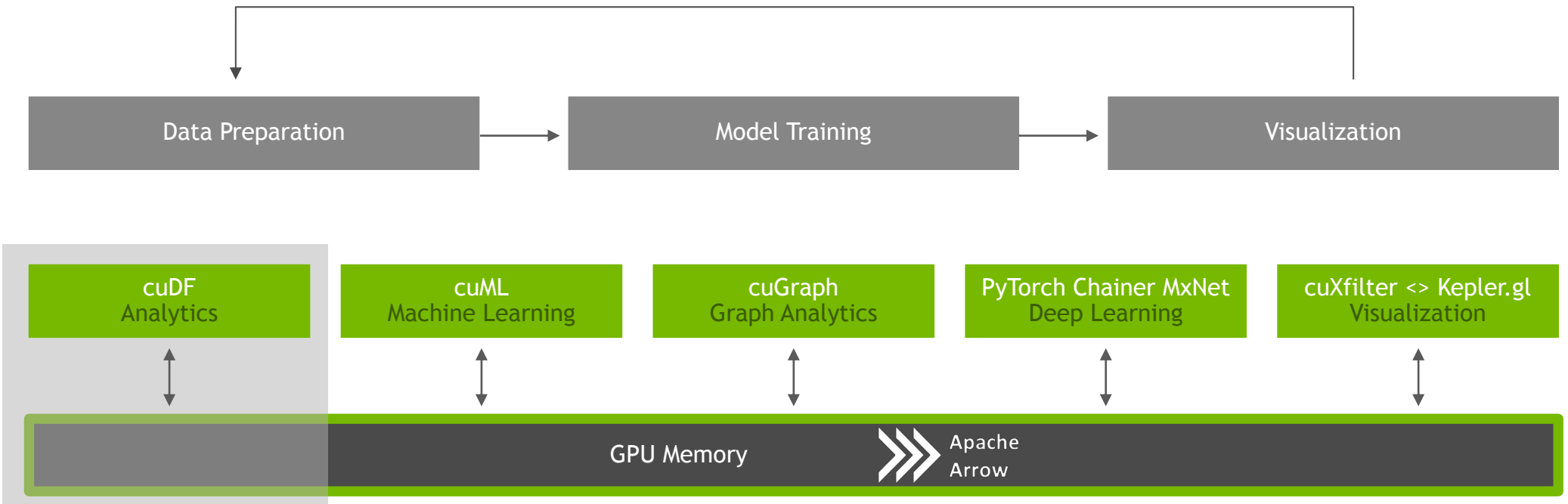


- https://github.com/rapidsai
- https://anaconda.org/rapidsai/
- https://pypi.org/project/cudf
- https://pypi.org/project/cuml
- https://pypi.org/project/cugraph (coming soon)

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

RAPIDS

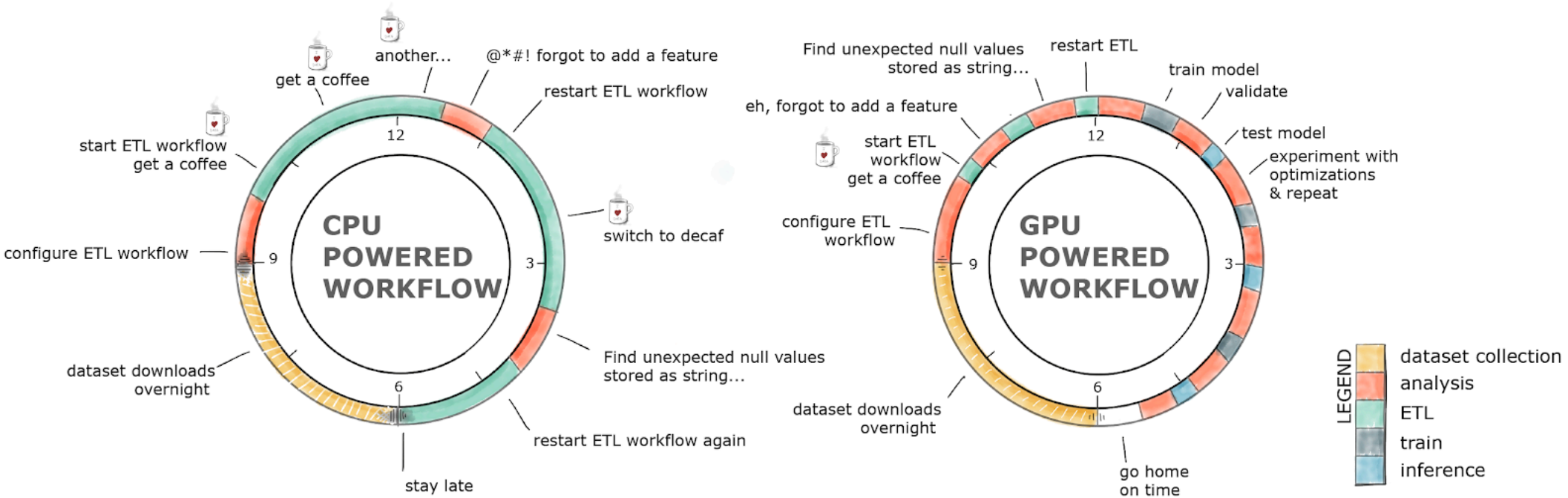
End to End Accelerate GPU Data Science



NVIDIA.

GPU-ACCELERATED ETL

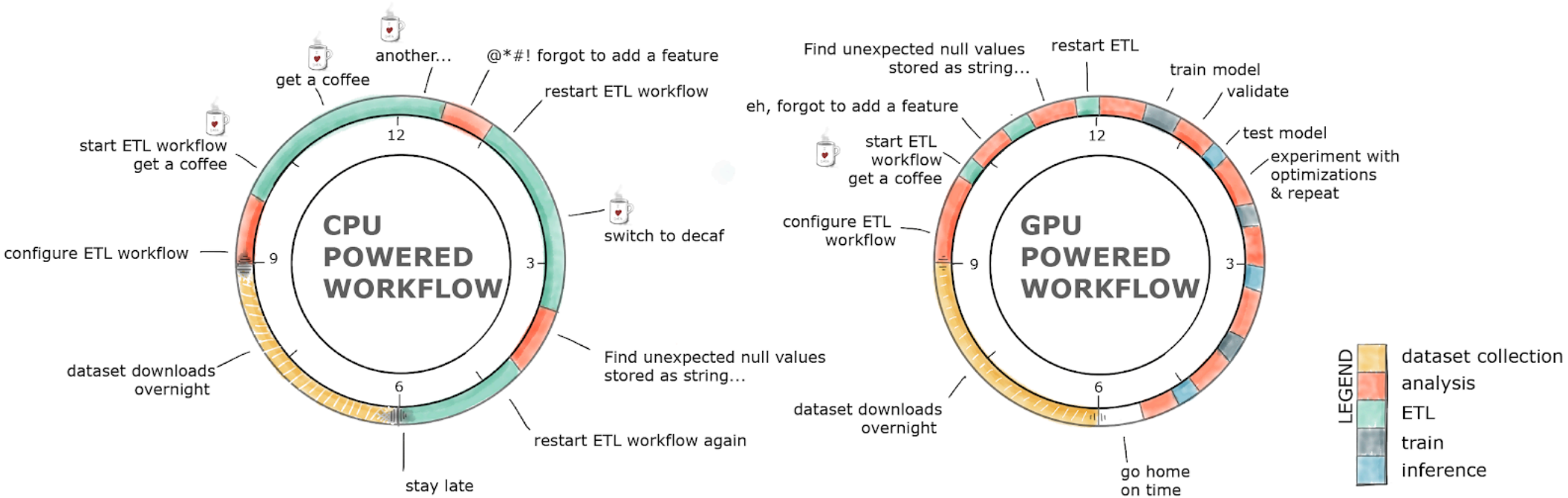
Is GPU-acceleration really needed?



NVIDIA.

GPU-ACCELERATED ETL

The average data scientist spends 90+% of their time in ETL as opposed to training models



CUDF

GPU DataFrame library

	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

- Apache Arrow data format
- Pandas-like API

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

NVIDIA.

CUDF

libcudf (CUDA C++)

- Low level library containing function implementations and C/C++ API
- Importing/exporting a GDF 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

cudf (Python)

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

NVIDIA.

CUDF

See Jake Hemstad's talk "RAPIDS CUDA DataFrame Internals for C++ Developers" on Wednesday at 10am

libcudf (CUDA C++)

- Low level library containing function implementations and C/C++ API
- Importing/exporting a GDF 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

cudf (Python)

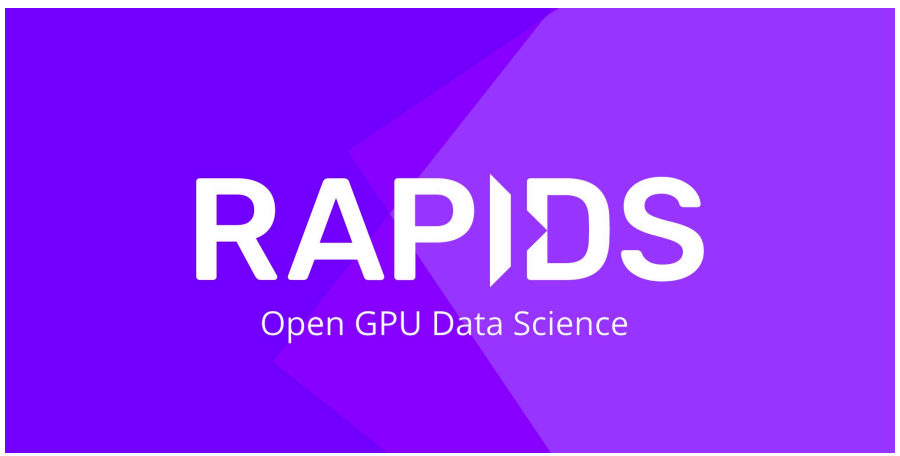
- A Python library for manipulating GPU DataFrames
- Python interface to libcudf library with additional functionality
- Creating GDFs from Numpy arrays, Pandas DataFrames, and PyArrow Tables
- JIT compilation of User-Defined Functions (UDFs) using Numba

NVIDIA.


LIVE DEMO!
(PRAY TO THE DEMO GODS)

CUDF

0.6 Release on Friday!



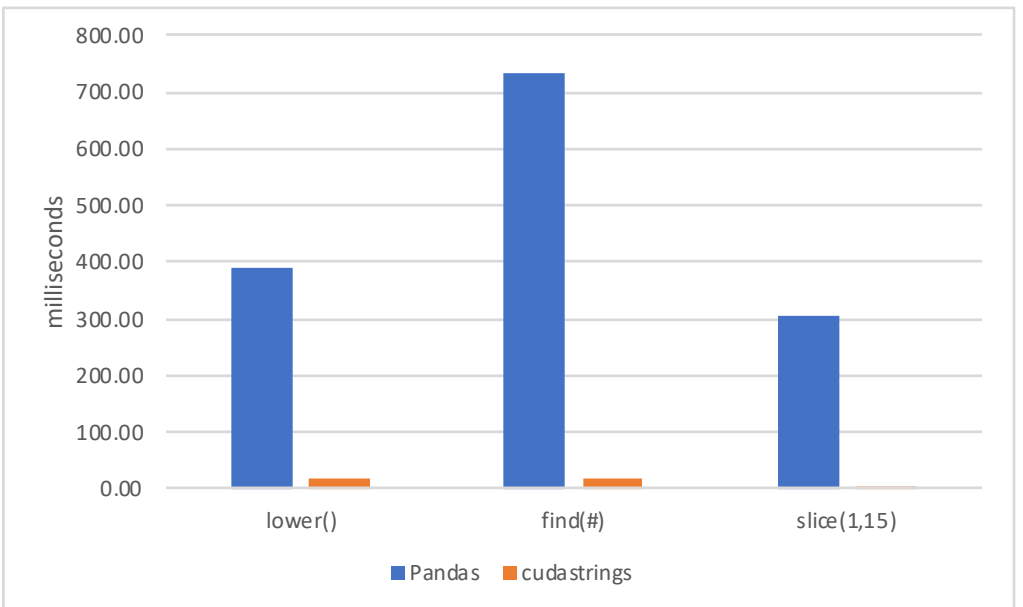
- Initial String Support!
- Near feature parity with Pandas on CSV Reader
- DLPack and __cuda_array_interface__ integration
- Huge API improvements for Pandas compatibility and enhanced multi-GPU capabilities via Dask
- Type-generic operation groundwork
- And more!

NVIDIA.

STRING SUPPORT

GPU-Accelerated string functions with a Pandas-like API

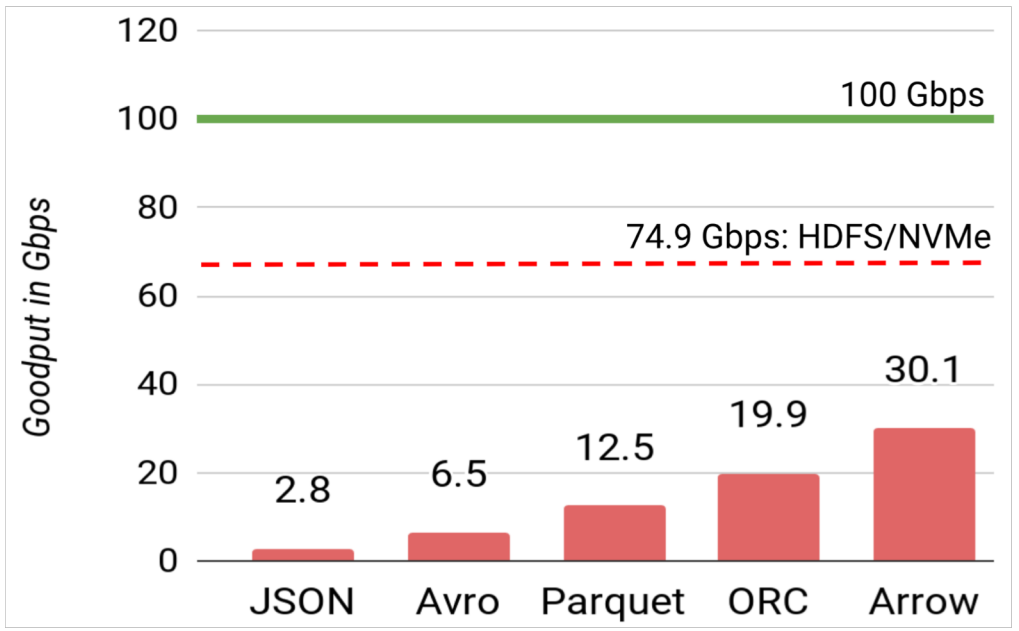
- API and functionality is following Pandas: https://pandas.pydata.org/pandas-docs/stable/api.html#string-handling
- Handles ingesting and exporting typical Python objects (Pandas series, Numpy arrays, PyArrow arrays, Python lists, etc.)
- Initial performance results:
  - lower(): ~22x speedup
  - find(): ~40x speedup
  - slice(): ~100x speedup



ACCELERATED DATA LOADING

CPUs bottleneck data loading in high throughput systems

- 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
- Decompression of the data will be GPU-accelerated as well!



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

NVIDIA.

INTEROPERABILITY WITH THE ECOSYSTEM

__cuda_array_interface__ and DLPack

NVIDIA.

PYTHON CUDA ARRAY INTERFACE

Interoperability for Python GPU Array Libraries

- The CUDA array interface is a standard format that describes a GPU array to allow sharing GPU arrays between different libraries without needing to copy or convert data

- Native ingest and export of __cuda_array_interface__ compatible objects via Numba device arrays in cuDF

- Numba, CuPy, and PyTorch are the first libraries to adopt the interface:
  - https://numba.pydata.org/numba-doc/dev/cuda/cuda_array_interface.html
  - https://github.com/cupy/cupy/releases/tag/v5.0.0b4
  - https://github.com/pytorch/pytorch/pull/11984

DLPACK

Interoperability with Deep Learning Libraries

- DLPack is an open-source memory tensor structure designed to allow sharing tensors between deep learning frameworks
- Currently supported by PyTorch, MXNet, and Chainer / CuPy
- cuDF supports ingesting and exporting column-major DLPack tensors
  - If you're interested in row-major tensor support please let us know!

NVIDIA.

DASK

What is Dask and why does RAPIDS use it for scaling out?

- Dask is a distributed computation scheduler built to scale Python workloads from laptops to supercomputer clusters.
- Extremely modular with scheduling, compute, data transfer, and out-of-core handling all being disjointed allowing us to plug in our own implementations.
- Can easily run multiple Dask workers per node to allow for an easier development model of one worker per GPU regardless of single node or multi node environment.

NVIDIA.

DASK

Scale up and out with cuDF

- Use cuDF primitives underneath in map-reduce style operations with the same high level API
- Instead of using typical Dask data movement of pickling objects and sending via TCP sockets, take advantage of hardware advancements using a communications framework called OpenUCX:
  - For intranode data movement, utilize NVLink and PCIe peer-to-peer communications
  - For internode data movement, utilize GPU RDMA over Infiniband and RoCE

http://www.openucx.org/

https://github.com/rapidsai/dask-cudf

NVIDIA.

DASK

Scale up and out with cuDF

- Use cuDF primitives underneath in map-reduce style operations with the same high level API
- Instead of using typical Dask data movement of pickling objects and sending via TCP sockets, take advantage of hardware advancements using a communications framework called OpenUCX:
  - For intranode data movement, utilize NVLink and PCIe peer-to-peer communications
  - For internode data movement, utilize GPU RDMA over Infiniband and RoCE

See Matt Rocklin's talk "Dask Extensions and New Developments with RAPIDS" next!

http://www.openucx.org/

https://github.com/rapidsai/dask-cudf

CUDF

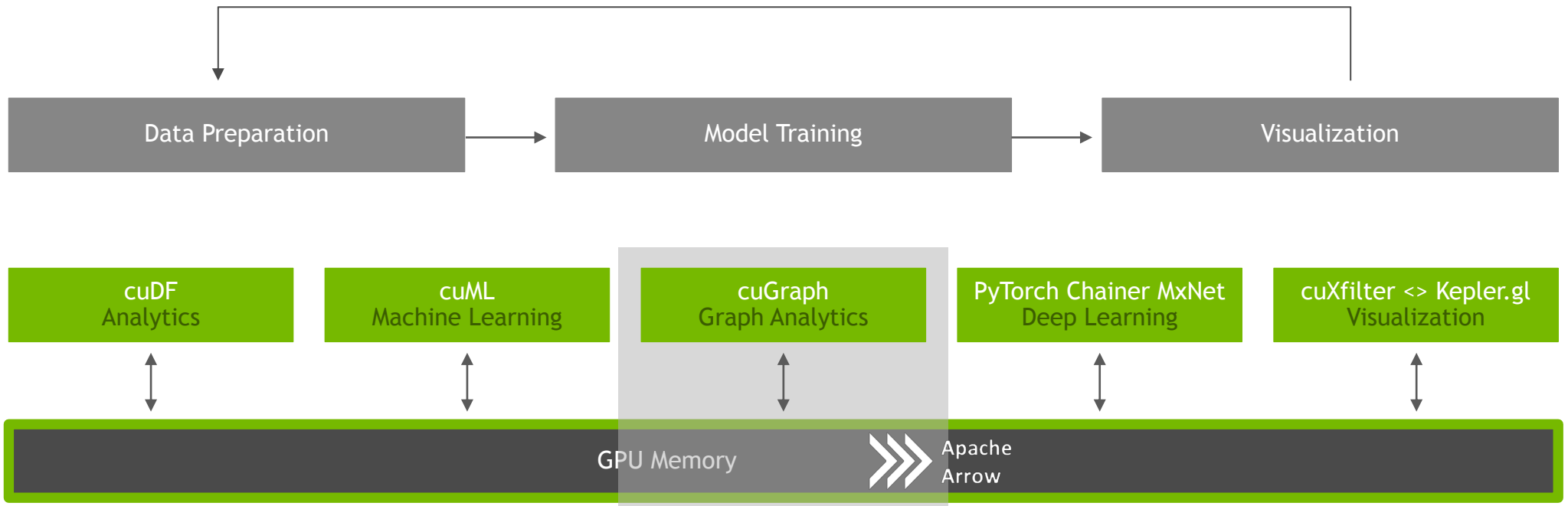
What's coming in 0.7+?

- Better User Experience
  - Migrating to using Cython exclusively for binding the Python to libcudf which allows for raising more intuitive and descriptive exceptions from the C++ API
  - Improve general exceptions and error handling in cuDF library for common issues such as driver / CUDA mismatches, out of memory errors, dtype mismatches, etc.
- More feature completeness in libcudf
  - Much of the functionality today lives purely in the Python library via just in time compiled kernels with Numba, we want to move this functionality to static compiled kernels in libcudf and expose usable C++ APIs for both end users and library builders
- Enhanced Multi-GPU and Multi-Node capabilities
  - Better Pandas API compatibility to integrate more into the Dask codebase

NVIDIA.

RAPIDS

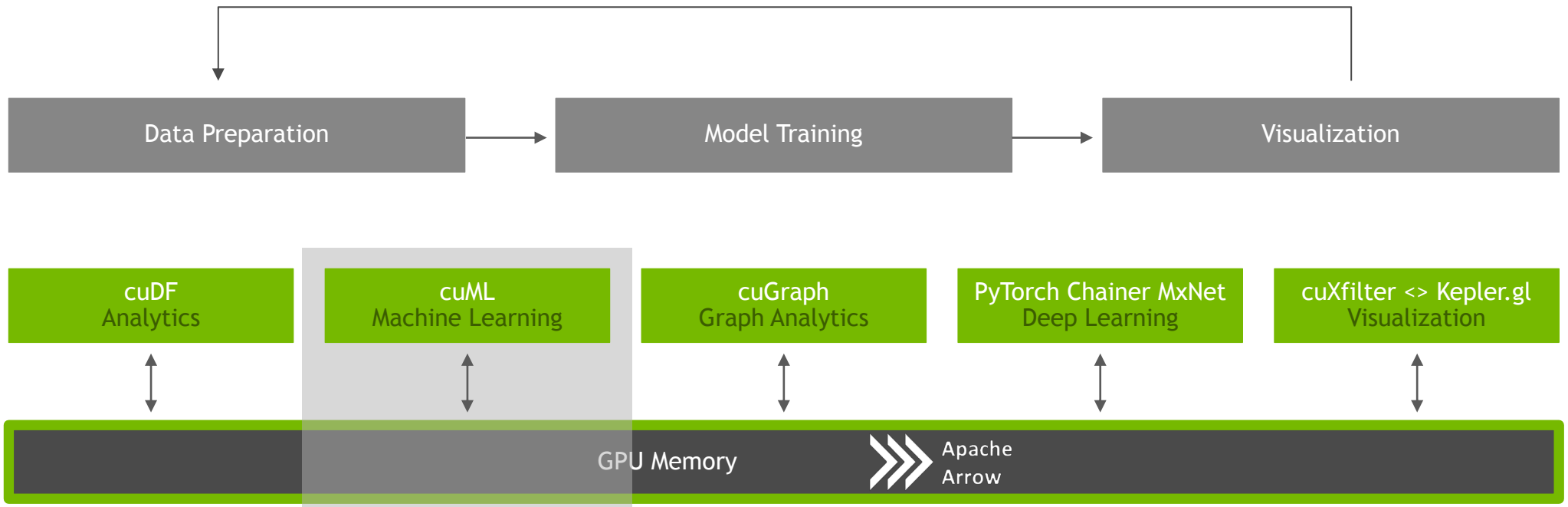
End to End Accelerate GPU Data Science



NVIDIA.

RAPIDS

End to End Accelerate GPU Data Science



NVIDIA.

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			

NVIDIA.

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			

NVIDIA.

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	■		

NVIDIA.

DASK

Scale up and out with cuML

- Native integration with Dask + cuDF
- Can easily use Dask workers to initialize NCCL for optimized gather / scatter operations
  - Example: this is how the dask-xgboost included in the container works for multi-GPU and multi-node, multi-GPU
- Provides easy to use, high level primitives for synchronization of workers which is needed for many ML algorithms

NVIDIA.

LOOKING TO THE FUTURE

JOIN THE MOVEMENT

Everyone Can Help!

APACHE ARROW

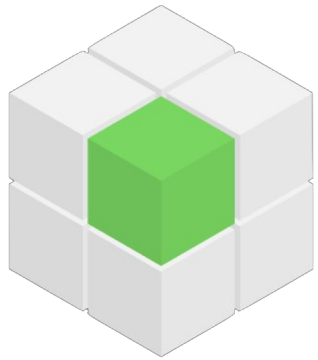
https://arrow.apache.org/

@ApacheArrow

RAPIDS

https://rapids.ai

@RAPIDSAI



GPU Open Analytics Initiative

http://gpuopenanalytics.com/

@GPUOAI

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

NVIDIA.

THANK YOU

Keith Kraus @keithjkraus
Dante Gama Dessavre @dante_dgd