Hewlett Packard Enterprise

SC24 ATLANTA NOV 17–22

Exploring Data at Scale with Arkouda: A Practical Introduction to Scalable Data Science

Ben McDonald, Software Engineer

Supercomputing 2024

Exploring Data at Scale with Arkouda

Outline

Challenge: Data scientists need to work interactively with massive-scale data sets

Introducing Arkouda: Python package designed for interactive, massive-scale data analysis

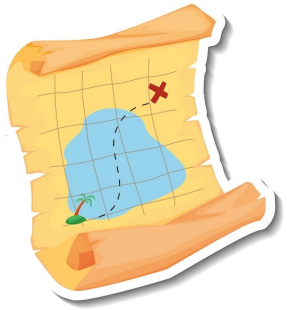
Demo

Recent advances: Rethinking Arkouda as a highly-extensible HPC framework

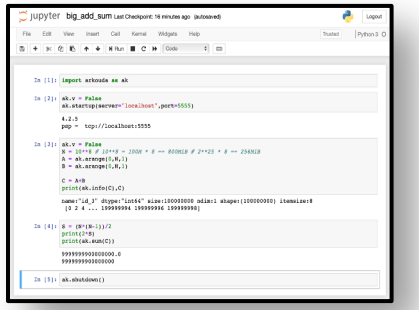
Exploring Data at Scale with Arkouda

Data Science

Working with big data is much like exploring uncharted territory



Jupyter notebook



Exploring Data at Scale with Arkouda

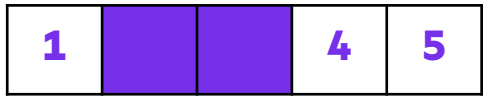
Workflows

- Data scientists are taught to work with data interactively, learning with their data as they go
  - In-memory, interactive analysis is preferred by many and how data science is taught (NumPy/Pandas)
  - Batch jobs are useful when workflow already developed, but doesn't provide the same intimacy
- To address the difficulty of working with big data, a typical workflow could be:
  1. Cut down to a subset of the data to fit in memory (**downsampling**)
  2. Work interactively on a single node to learn about the subset data (NumPy/Pandas)
  3. Take what has been learned and send that off in a batch job on full data (Spark, etc.)

Exploring Data at Scale with Arkouda

The Streetlight Effect

Faced with the unknown (downsampling), data scientists can suffer from the **“lighthouse effect”**



Exploring Data at Scale with Arkouda

Workflows

Without Arkouda

- To address the difficulty of working with big data, a typical workflow could be:
  1. Cut down to a subset of the data to fit in memory (**downsampling**)
  2. Work interactively on a single node to learn about the data (NumPy, Pandas, etc.)
  3. Take what has been learned and send that off in a batch job (Spark, etc.)
- But what about the outliers? What about the streetlight effect?

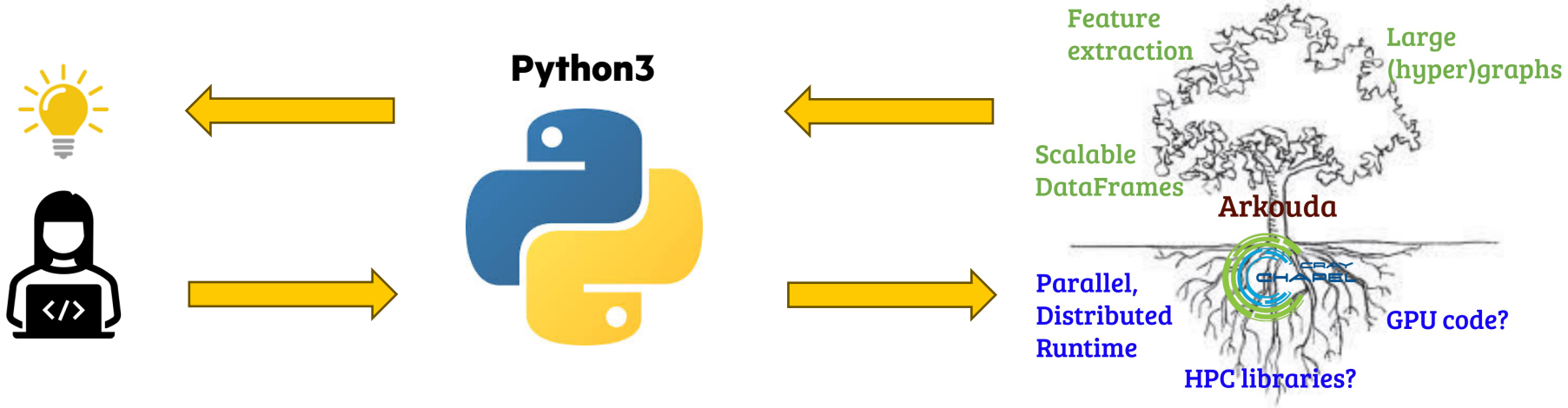
With Arkouda

1. Work **interactively** with full data set on as many nodes as needed (Arkouda)
2. Pass subset of data to NumPy/Pandas to work with as usual (**upside-downsampling**)
3. Take what has been learned and send that off in a batch job (Spark, etc.)

Data Science Beyond the Laptop

Data sets today

Python must scale **beyond the laptop**, without sacrificing **interactivity**



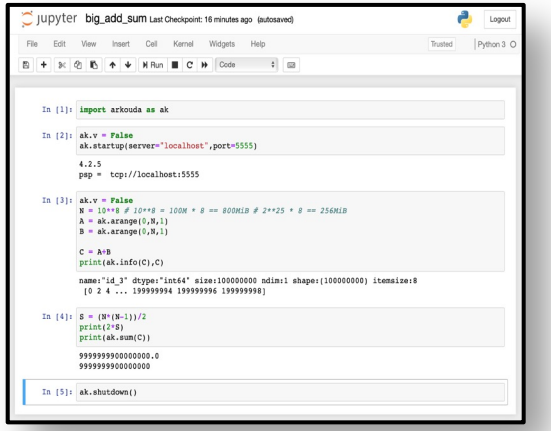
Data Science Beyond the Laptop

Arkouda

Interactivity

Arkouda Client

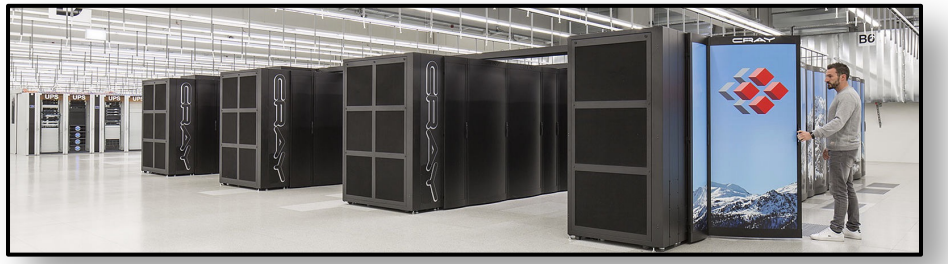
(written in Python)



Scalability

Arkouda Server

(written in Chapel)



Exploring Data at Scale with Arkouda

Demo and Following along…

Installation

- To install Arkouda, see https://bears-r-us.github.io/arkouda/setup/install_menu.html

Docker Containers

- https://github.com/Bears-R-Us/arkouda-contrib/tree/main/arkouda-docker

Tutorial and Codespace

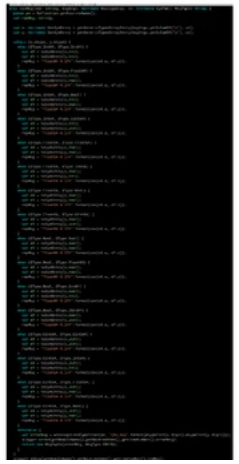
- https://github.com/bmcdonald3/arkouda-codespace

Exploring Data at Scale with Arkouda

Recent Development

- Interest in Arkouda outside of data science community has led to rearchitecting
- Arkouda has traditionally been thought of as “NumPy for HPC”
  - Rethinking Arkouda as a general framework for rapidly developing HPC-ready Python packages

Before (97 lines)



Today (7 lines)

```
@arkouda.registerCommand()
proc cov(const ref x: [?dx] ?tx,
         const ref y: [?dy] ?ty): real throws {
  if dx.shape != dy.shape then
    throw new Error("x and y must have the same shape");

  const mx = mean(x),
    my = mean(y);

  return (+ reduce ((x:real - mx) * (y:real - my))) / (dx.size - 1):real
}
```

- What previously required 97 lines in Arkouda can now be written as 7
  - Any Chapel function can be called from Python by adding a ‘registerCommand’ annotation

Questions?

Data Science Beyond the Laptop

Arkouda

An open-source Python package providing interactive data analytics at supercomputing scale.

Transform the way you work with big data;
massive computation within the human thought loop

EASY TO USE

Provides an API data scientists are familiar with based on Pandas/NumPy

FAST & SCALABLE

Sorting 256 TiB of data on 8,000 Nodes within seconds

EXTENSIBLE & CUSTOMIZABLE

Highly extensible ecosystem allows rapid feature development and broad project collaboration

POWERED BY CHAPEL

Powered by a parallel distributed server written in Chapel

Data Science Beyond the Laptop

The Chapel Programming Language

From laptops to supercomputers, Chapel makes parallel programming more productive.

EASY TO USE

Supports code as approachable as Python and flexible as C++

>>>

Leverage the parallel power of your hardware quickly.

FAST & SCALABLE

Scales to millions of cores with performance that rivals MPI

>>>

Scale your applications with ease.

PORTABLE

Executes on: HPE Apollo, HPE Cray EX, HPE Superdome Flex, Linux/*nix systems, Mac, NVIDIA and AMD GPUs

>>>

Write your code once and run it anywhere.

GPU-READY

Supports high-level, vendor-neutral GPU programming without language extensions

>>>

Unlock the power of GPUs for parallel computing.

OPEN SOURCE

Developed by HPE on GitHub in collaboration with the open-source community

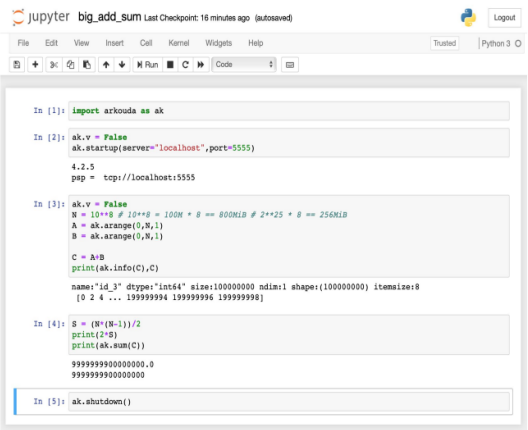
>>>

Join a growing community of Chapel users and developers!

Data Science Beyond the Laptop

Arkouda

Python3 Client



ZMQ Socket

Chapel Server

Code Modules

Distributed Object Store

Platform

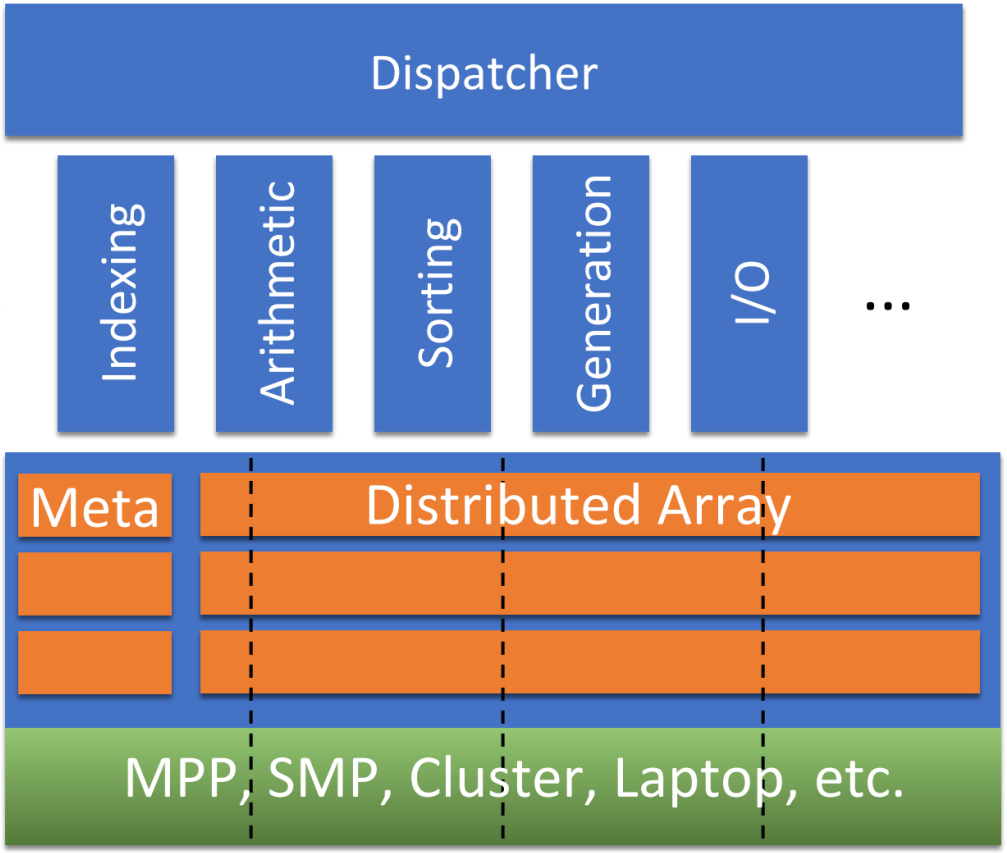


Image from Bill Reuss's 2020 Chapel keynote

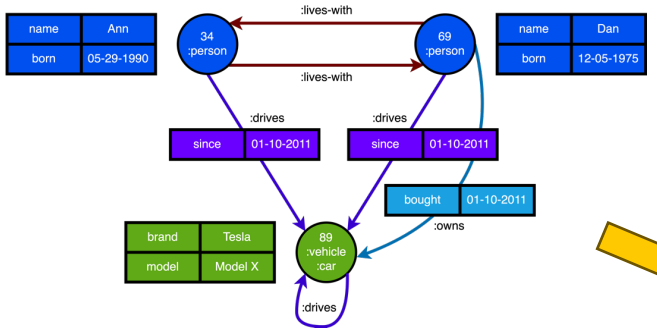
Data Science Beyond the Laptop

The Arkouda Ecosystem

id	label	name	born	brand	model
34					
69					
89					
89					

src id	dst id	relationship	since	bought
34	69	lives-with	NULL	NULL
69	34	lives-with	NULL	NULL
34	89	drives	2011	NULL
69	89	drives	2011	NULL
69	89	owns	NULL	2011
89	89	drives	NULL	NULL

Arachne



bfs_layers()
subgraph_isomorphism()
square_counting()
subgraph_view()

NJIT New Jersey Institute of Technology

Data Science Beyond the Laptop

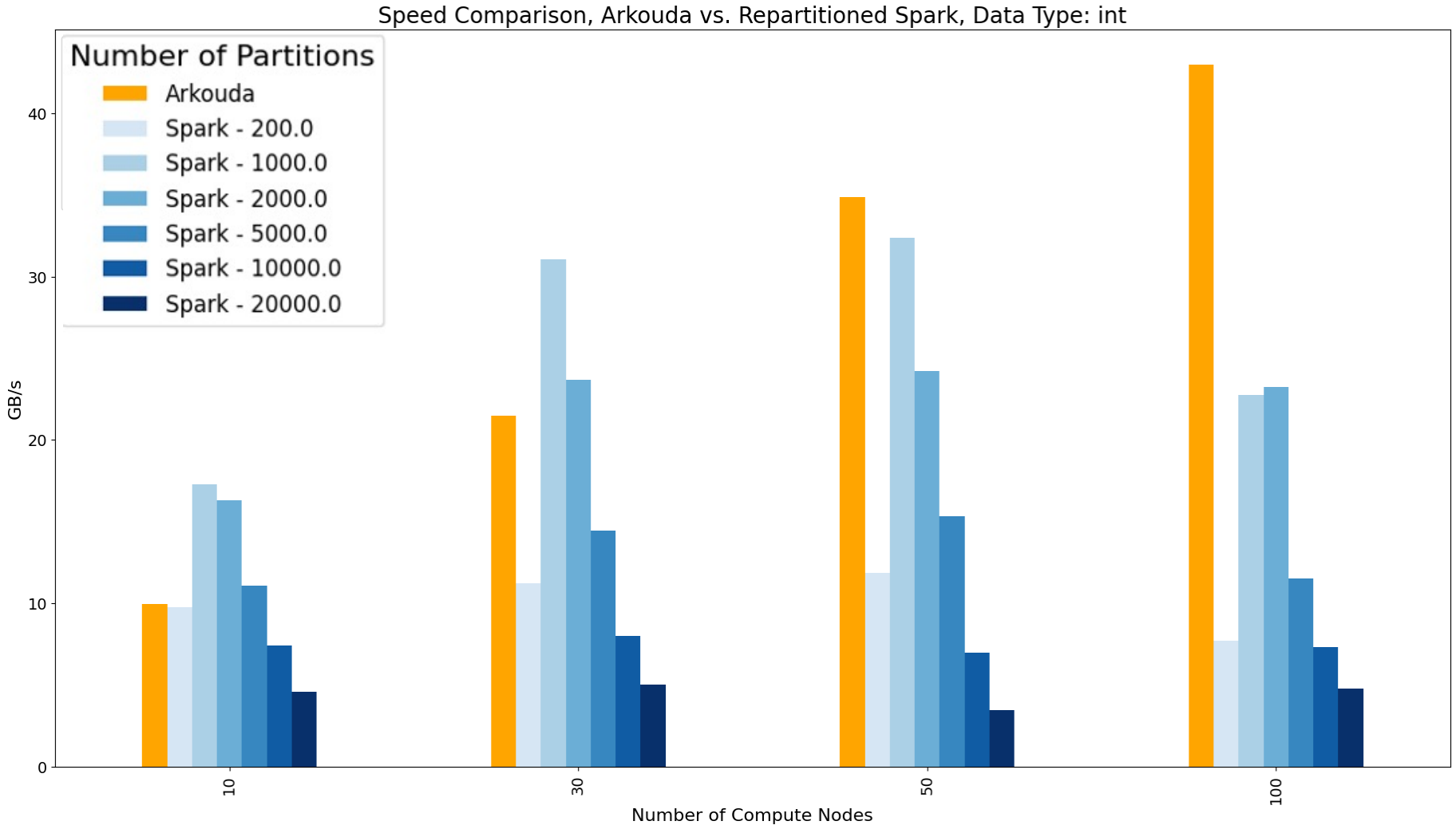
Data Science

Data science requires an intimacy with data reached through **interactive exploration**

Regardless of data **quantity**, **quality** requires **scalable** workflows on the complete dataset

Data Science Beyond the Laptop

Parquet read + GroupBy performance vs Spark



Data Science Beyond the Laptop

Arkouda Sort Performance