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

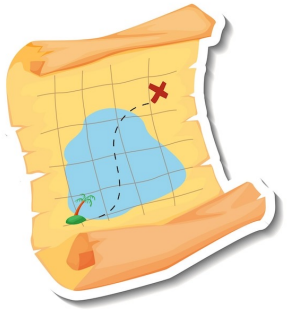
Demo

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

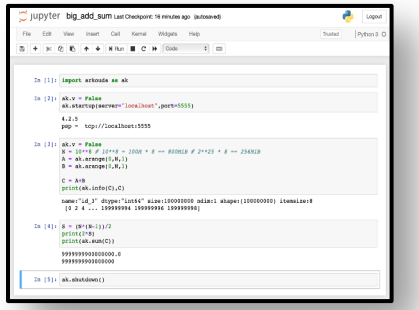
Exploring Data at Scale with Arkouda

Data Science

Working with big data is much like exploring uncharted territory



Jupyter notebook



Exploring Data at Scale with Arkouda

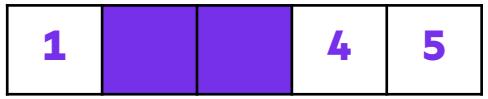
Workflows

- 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?

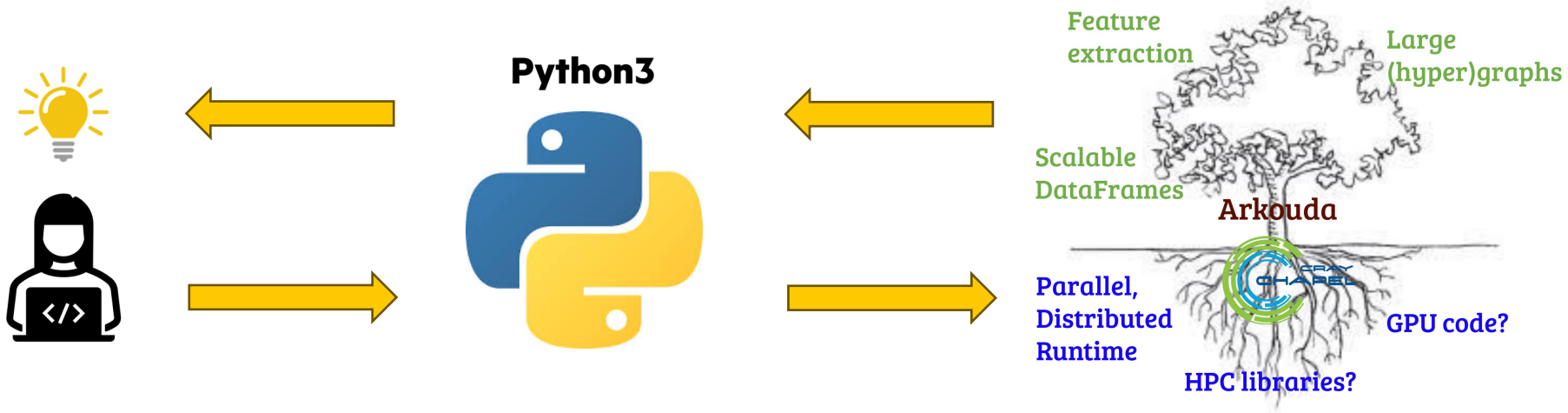
With Arkouda

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**

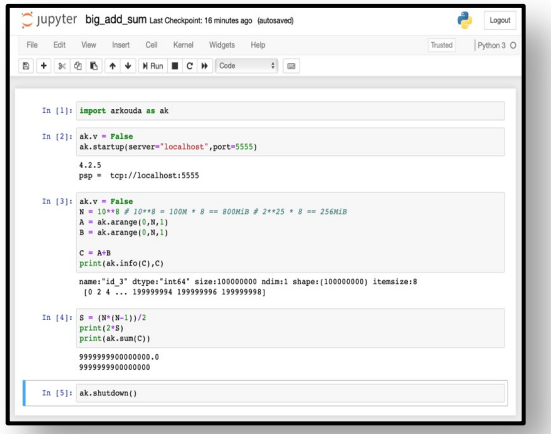


Data Science Beyond the Laptop

Arkouda

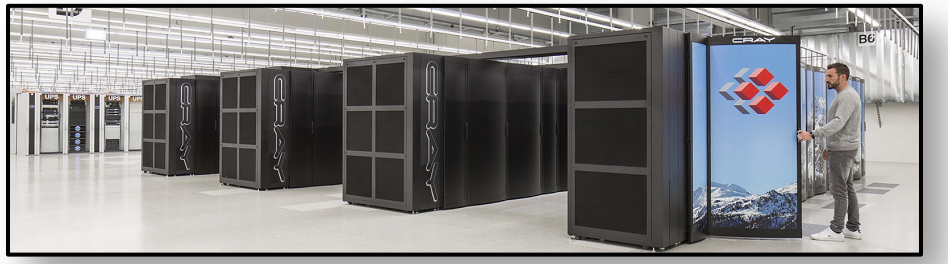
Interactivity

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

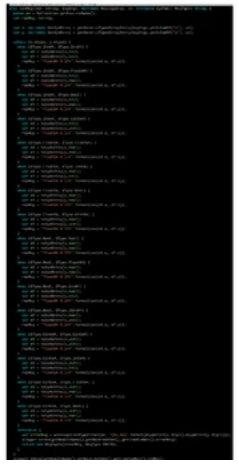
View the video that was embedded here at https://www.youtube.com/watch?v=eSuyl9ogDfI

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?