Hewlett Packard Enterprise

Data Science Beyond the Laptop: Handling Data of Any Size with Arkouda

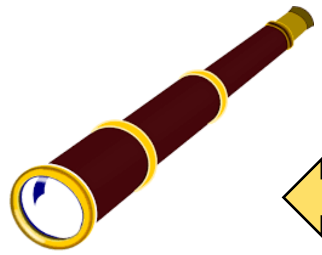
Ben McDonald, Michelle Strout, Sarah Coghlan, Oilver Alvarado Rodriguez

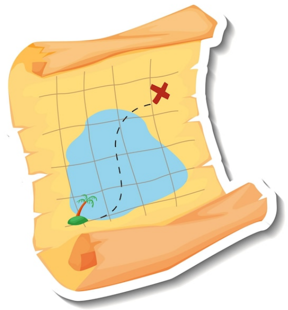
ChapelCon June 5, 2024

Data Science Beyond the Laptop

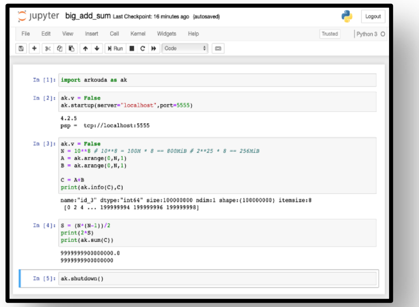
Data Science

Data Scientists are modern-day explorers uncovering previously unimagined insights





Jupyter notebook



jupyter big_add_sum Last Checkpoint: 16 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

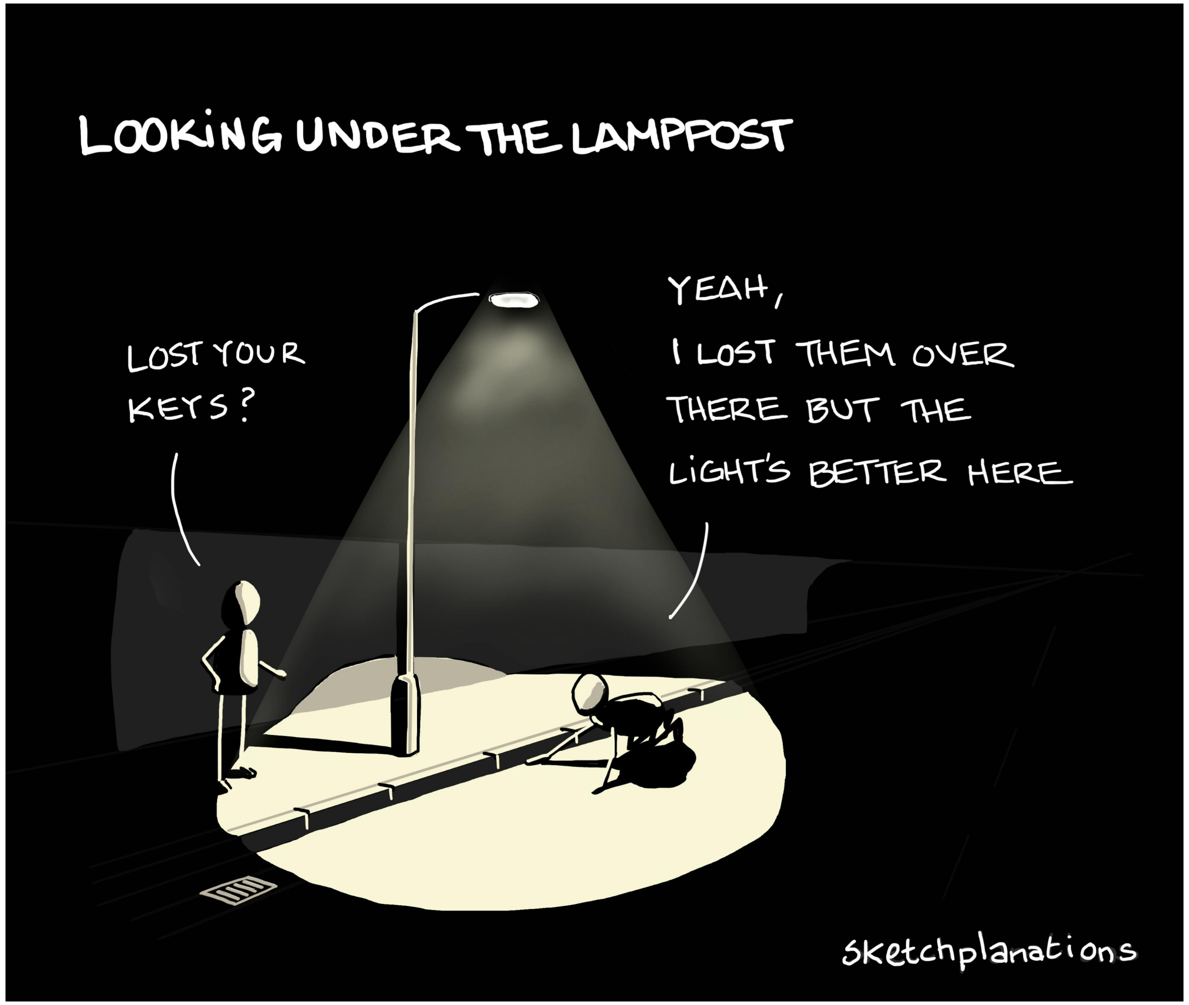
```
In [1]: import arkouda as ak

In [2]: ak.v = False
        ak.startup(server="localhost",port=5555)

        4.2.5
        psp =  tcp://localhost:5555

In [3]: ak.v = False
        N = 10**8 # 10**8 = 100M * 8 == 800MiB # 2**25 * 8 == 256MiB
        A = ak.arange(0,N,1)
        B = ak.arange(0,N,1)

        C = A+B
        print(ak.info(C),C)

        name:"id_3" dtype:"int64" size:100000000 ndim:1 shape:(100000000) itemsize:8
         [0 2 4 ... 199999994 199999996 199999998]

In [4]: S = (N*(N-1))/2
        print(2*S)
        print(ak.sum(C))

        9999999900000000.0
        9999999900000000

In [5]: ak.shutdown()
```

Data Science Beyond the Laptop

The Streetlight Effect

Faced with the unknown, data scientists as well as explorers can suffer from the **“lighthouse effect”**

1	🟪	🟪	4	5


LOOKING UNDER THE LAMPPOST
LOST YOUR KEYS?
YEAH, I LOST THEM OVER THERE BUT THE LIGHT'S BETTER HERE
sketchplanations

Data Science Beyond the Laptop

Data sets today

What if we could work with **massive-scale** data in its entirety **without compromising interactivity**?

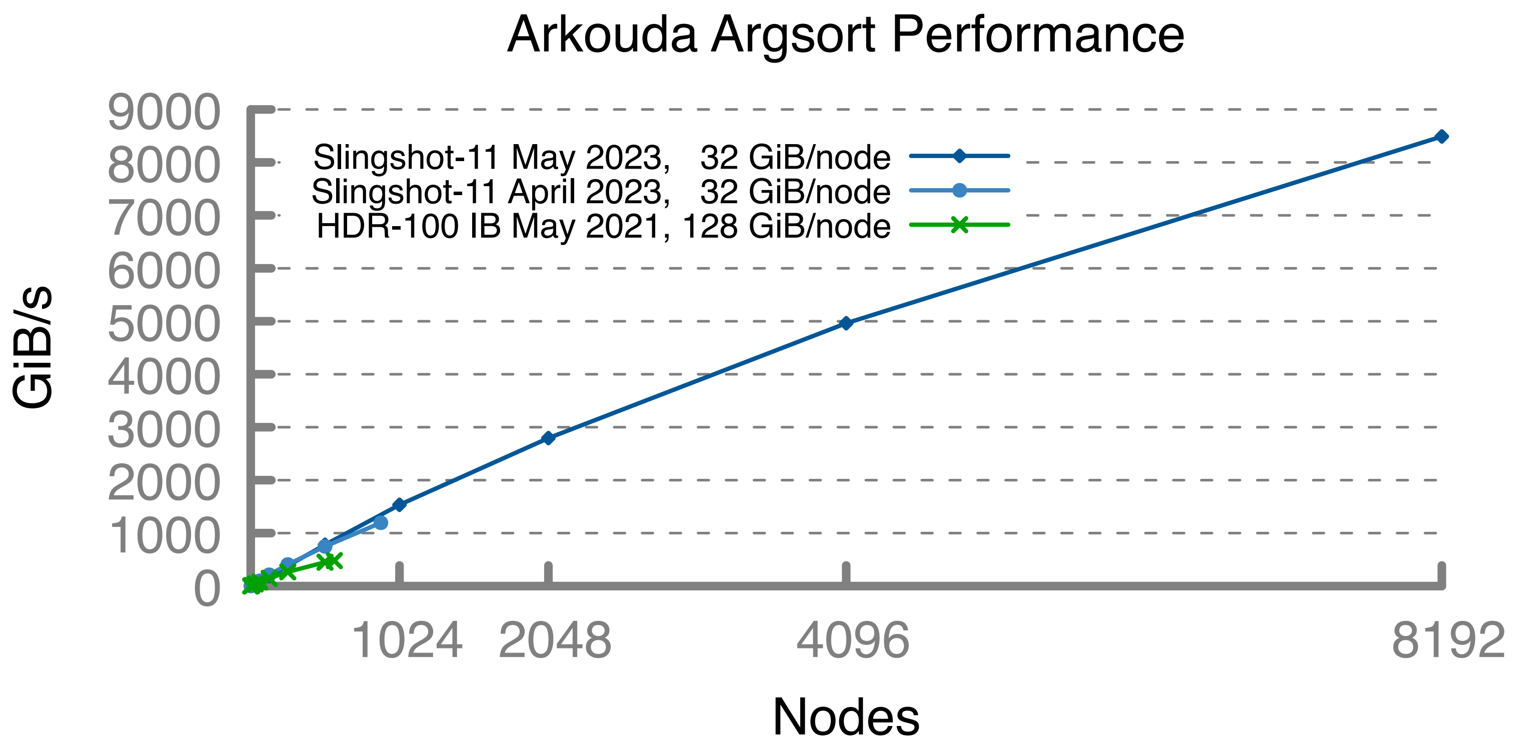
Arkouda (αρκούδα)

Data Science Beyond the Laptop

Arkouda Performance

HPE Cray EX (May 2023)

- Slingshot-11 network (200 Gb/s)
- 8192 compute nodes
- 256 TiB of 8-byte values
- **~8500 GiB/s (~31 seconds)**



- "…solving problems in a matter of seconds, rather than days…" – Tess Hayes, Bytoa

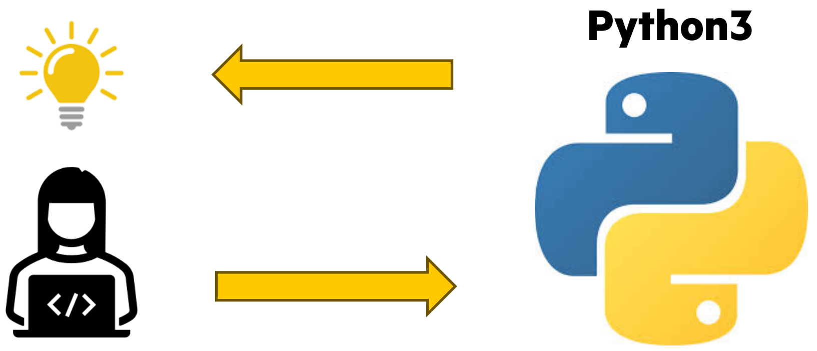
Data Science Beyond The Laptop

- Demands of data science today
- What is Arkouda?
- What is Chapel?
- The Arkouda ecosystem
- Conclusion
- Tutorial…
- The future of Arkouda

Data Science Beyond the Laptop

Data sets today

- Data scientists are drawn to Python for its interactivity
  - Code executes through the REPL (read, evaluate, print, loop)
  - Operations complete within human thought-loop
- **Data science demands interactivity…**



Data Science Beyond the Laptop

The Python Landscape

Data Science Python is only **syntactically Python**

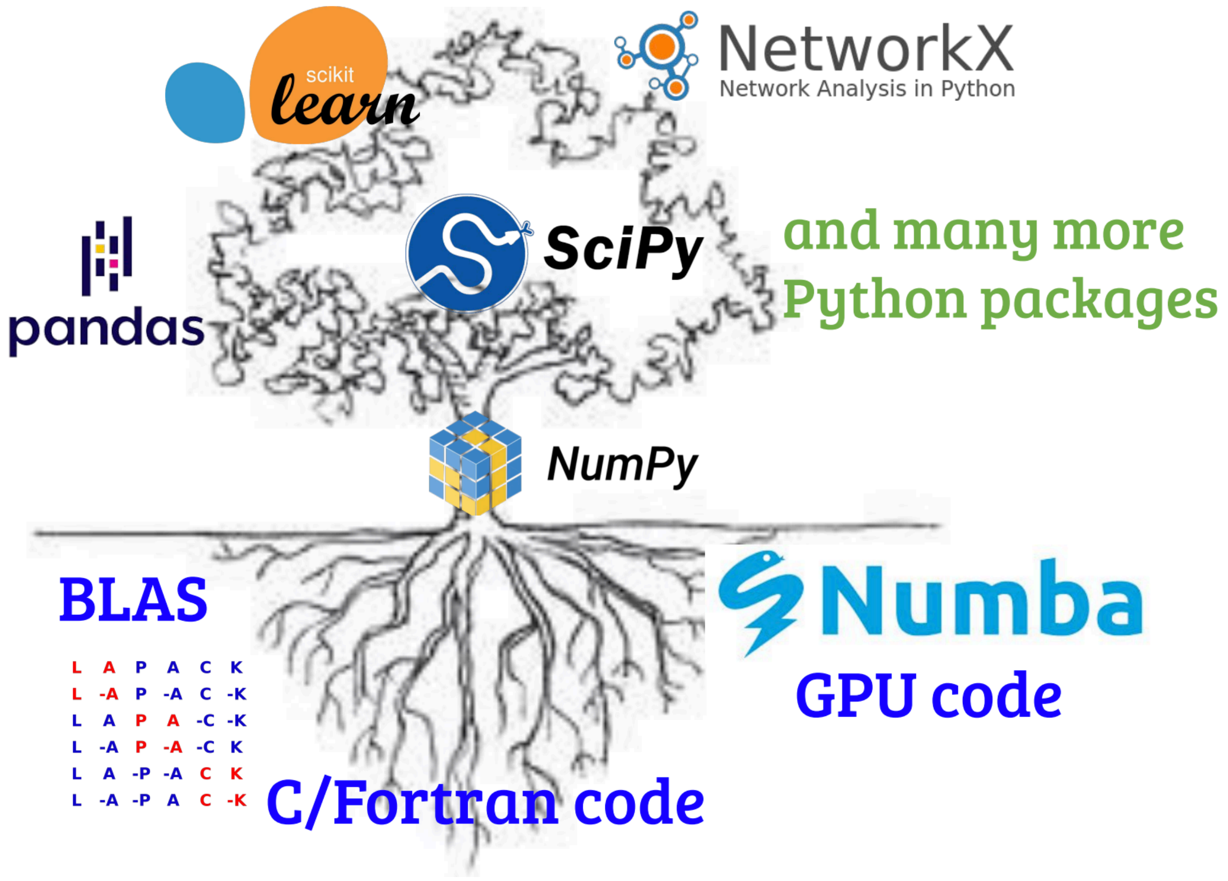
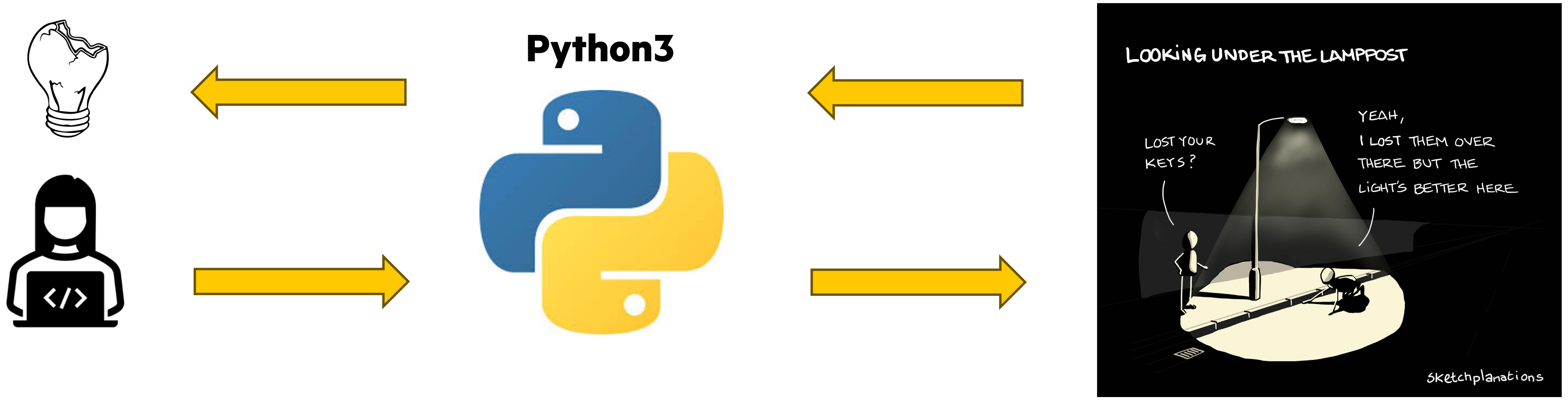


Image from Bill Reuss's 2020 Chapel keynote

Data Science Beyond the Laptop

Data sets today

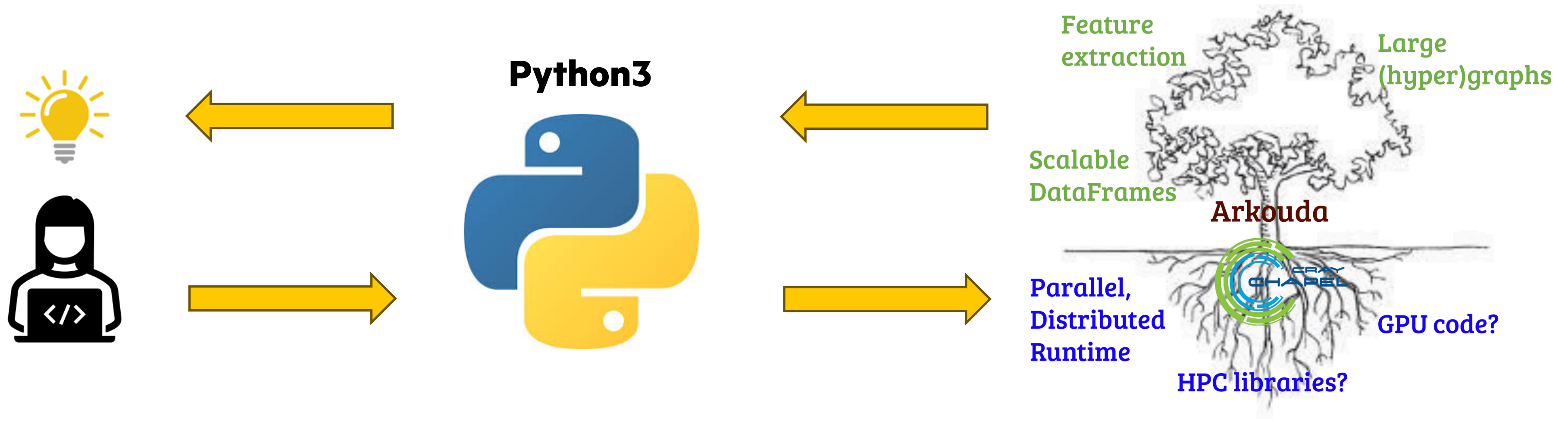
- Data sets have outgrown typical computers
  - Unbiased sampling is difficult
  - Unbiased sampling can eliminate rare and high-order effects
- **Data science demands scalability…**


Python3
LOOKING UNDER THE LAMPPOST
LOST YOUR KEYS?
YEAH, I LOST THEM OVER THERE BUT THE LIGHT'S BETTER HERE
sketchplanations

Data Science Beyond the Laptop

Data sets today

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

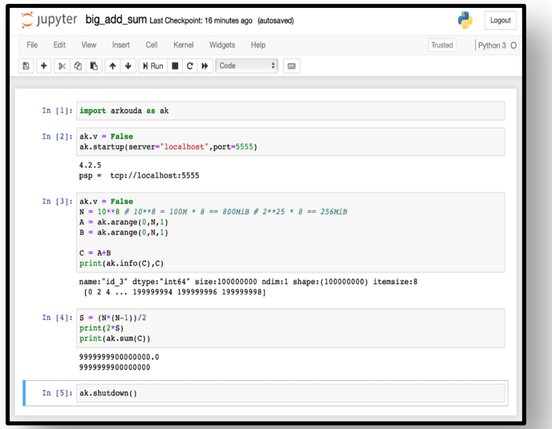


Data Science Beyond the Laptop
Arkouda

Interactivity

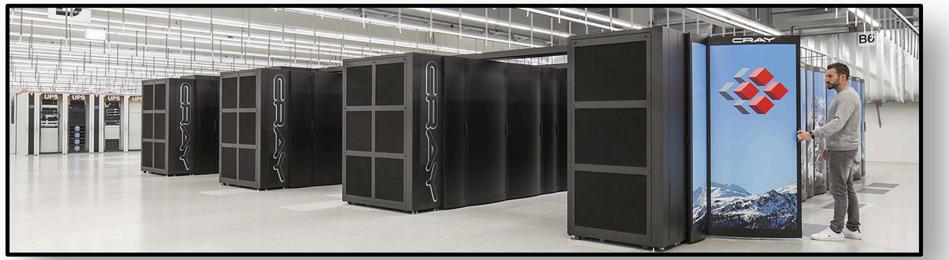
Scalability

Arkouda Client
(written in Python)


jupyter big_add_sum Last Checkpoint: 16 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [1]: import arkouda as ak

In [2]: ak.v = False
        ak.startup(server="localhost",port=5555)

        4.2.5
        psp =  tcp://localhost:5555

In [3]: ak.v = False
        N = 10**8 # 10**8 = 100M * 8 == 800MiB # 2**25 * 8 == 256MiB
        A = ak.arange(0,N,1)
        B = ak.arange(0,N,1)

        C = A+B
        print(ak.info(C),C)

        name:"id_3" dtype:"int64" size:100000000 ndim:1 shape:(100000000) itemsize:8
         [0 2 4 ... 199999994 199999996 199999998]

In [4]: S = (N*(N-1))/2
        print(2*S)
        print(ak.sum(C))

        9999999900000000.0
        9999999900000000

In [5]: ak.shutdown()
```

Arkouda Server
(written in Chapel)



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

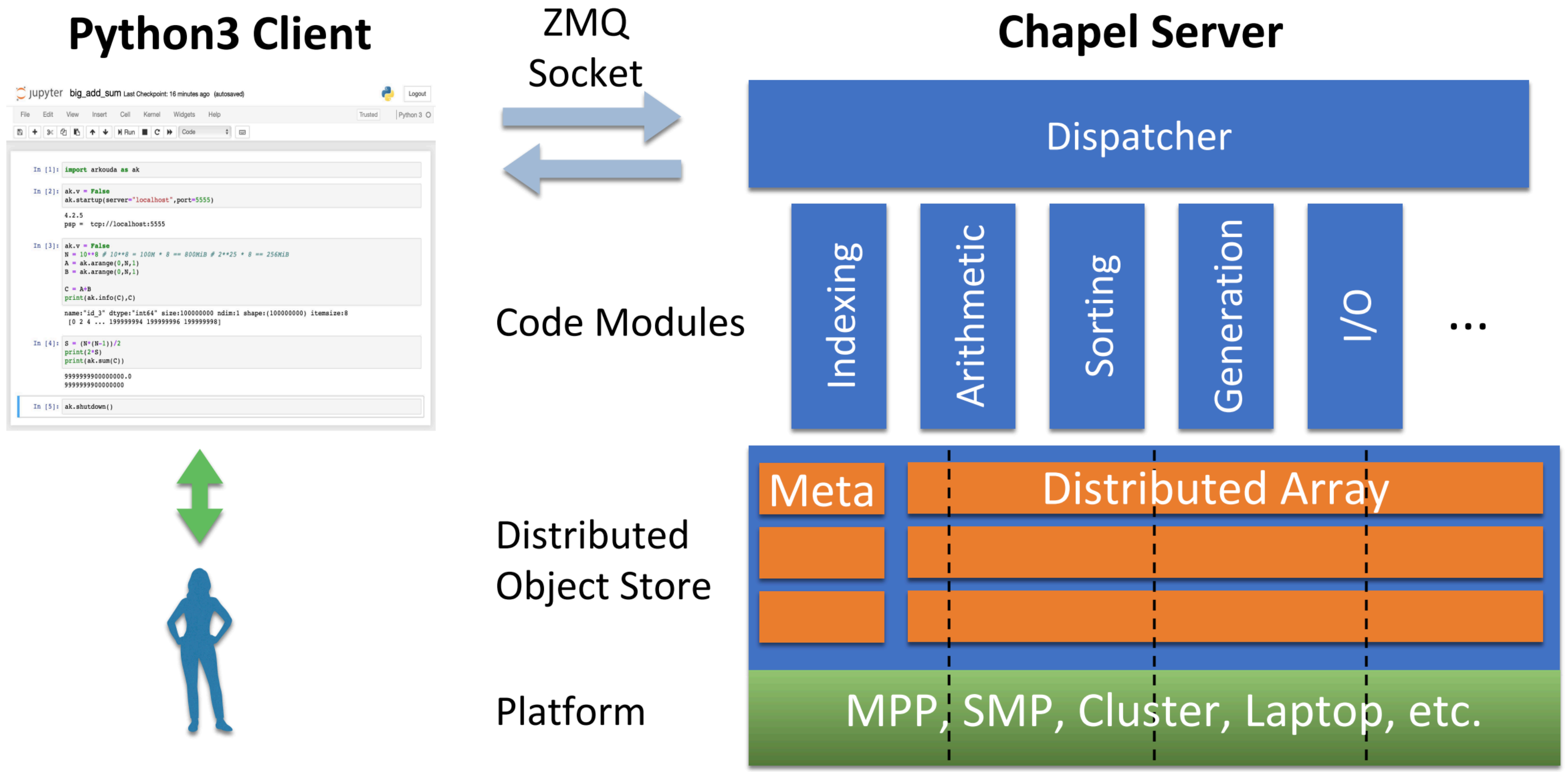
Developed by HPE on GitHub in collaboration with the open-source community

>>>

Join a growing community of Chapel users and developers!

Data Science Beyond the Laptop

Arkouda

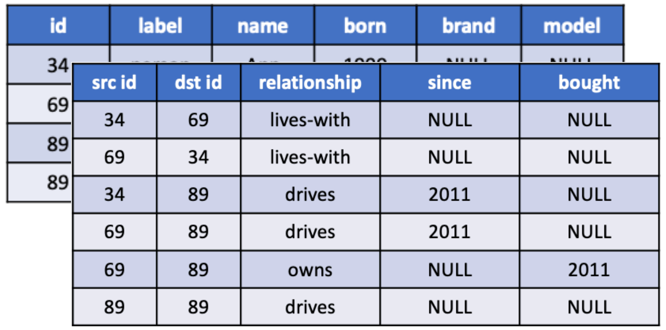


```
In [1]: import arkouda as ak

In [2]: ak.v = False
        ak.startup(server="localhost",port=5555)

        4.2.5
        psp =  tcp://localhost:5555

In [3]: ak.v = False
        N = 10**8 # 10**8 = 100M * 8 == 800MiB # 2**25 * 8 == 256MiB
        A = ak.arange(0,N,1)
        B = ak.arange(0,N,1)

        C = A+B
        print(ak.info(C),C)

        name:"id_3" dtype:"int64" size:100000000 ndim:1 shape:(100000000) itemsize:8
         [0 2 4 ... 199999994 199999996 199999998]

In [4]: S = (N*(N-1))/2
        print(2*S)
        print(ak.sum(C))

        9999999900000000.0
        9999999900000000

In [5]: ak.shutdown()
```

Image from Bill Reuss's 2020 Chapel keynote

Data Science Beyond the Laptop

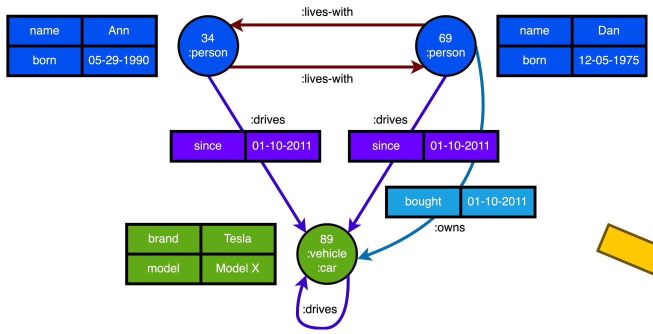
The Arkouda Ecosystem



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



NJIT
New Jersey Institute of Technology

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

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
Demo

Learning Objective

- How to use Arkouda
  - Launch a server
  - Connect to a server
  - Create arrays
- The Arkouda API
  - Supported file formats
  - Exploratory data analysis

Follow Along with an Arkouda Codespace

- https://github.com/bmcdonald3/chapelcon-2024-arkouda

Hewlett Packard Enterprise

What’s in Store for Arkouda?

Ben McDonald, Michelle Strout, Sarah Coghlan, Oilver Alvarado Rodriguez

ChapelCon June 5, 2024

What’s in Store for Arkouda?

Next Steps

Developer Experience

- Efforts underway to make “Arkouda code” look like “Chapel code” (eliminating boilerplate)

Resiliency

- Plans to improve server recoverability and improve server resiliency

User Experience

- Considering non-blocking interaction with Arkouda via a messaging queue to improve responsiveness

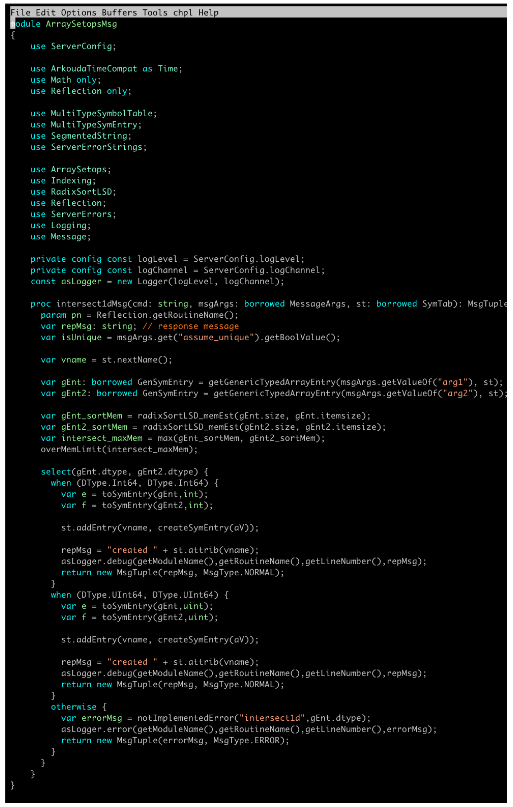
Modernization

- Working towards a web summary interface to provide detailed server information in real time

What’s in Store for Arkouda?

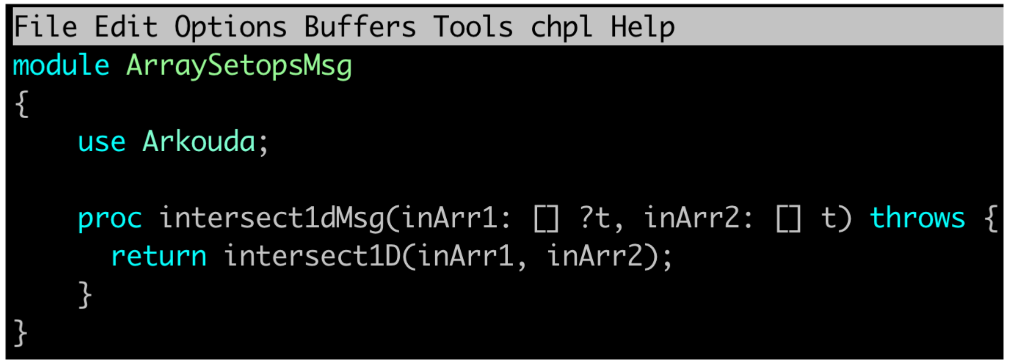
Boilerplate code

- Boilerplate required to Add Arkouda functionality today
  - 85 lines of unnecessary boilerplate code
  - Repetition of code for each Arkouda supported type (int, uint, bool, etc.)



```
module ArraySetopsMsg
{
    use ServerConfig;

    use ArkoudaTimeCompat as Time;
    use Math only;
    use Reflection only;

    use MultiTypeSymbolTable;
    use MultiTypeSymEntry;
    use SegmentedString;
    use ServerErrorStrings;

    use ArraySetops;
    use Indexing;
    use RadixSortLSD;
    use Reflection;
    use ServerErrors;
    use Logging;
    use Message;

    private config const logLevel = ServerConfig.logLevel;
    private config const logChannel = ServerConfig.logChannel;
    const asLogger = new Logger(logLevel, logChannel);

    proc intersect1dMsg(cmd: string, msgArgs: borrowed MessageArgs, st: borrowed SymTab): MsgTuple
      param pn = Reflection.getRoutineName();
      var repMsg: string; // response message
      var isUnique = msgArgs.get("assume_unique").getBoolValue();

      var vname = st.nextName();

      var gEnt: borrowed GenSymEntry = getGenericTypedArrayEntry(msgArgs.getValueOf("arg1"), st);
      var gEnt2: borrowed GenSymEntry = getGenericTypedArrayEntry(msgArgs.getValueOf("arg2"), st);

      var gEnt_sortMem = radixSortLSD_memEst(gEnt.size, gEnt.itemsize);
      var gEnt2_sortMem = radixSortLSD_memEst(gEnt2.size, gEnt2.itemsize);
      var intersect_maxMem = max(gEnt_sortMem, gEnt2_sortMem);
      overMemLimit(intersect_maxMem);

      select(gEnt.dtype, gEnt2.dtype) {
        when (DType.Int64, DType.Int64) {
          var e = toSymEntry(gEnt,int);
          var f = toSymEntry(gEnt2,int);

          st.addEntry(vname, createSymEntry(aV));

          repMsg = "created " + st.attrib(vname);
          asLogger.debug(getModuleName(),getRoutineName(),getLineNumber(),repMsg);
          return new MsgTuple(repMsg, MsgType.NORMAL);
        }
        when (DType.UInt64, DType.UInt64) {
          var e = toSymEntry(gEnt,uint);
          var f = toSymEntry(gEnt2,uint);

          st.addEntry(vname, createSymEntry(aV));

          repMsg = "created " + st.attrib(vname);
          asLogger.debug(getModuleName(),getRoutineName(),getLineNumber(),repMsg);
          return new MsgTuple(repMsg, MsgType.NORMAL);
        }
        otherwise {
          var errorMsg = notImplementedError("intersect1d",gEnt.dtype);
          asLogger.error(getModuleName(),getRoutineName(),getLineNumber(),errorMsg);
          return new MsgTuple(errorMsg, MsgType.ERROR);
        }
      }
    }
}
```

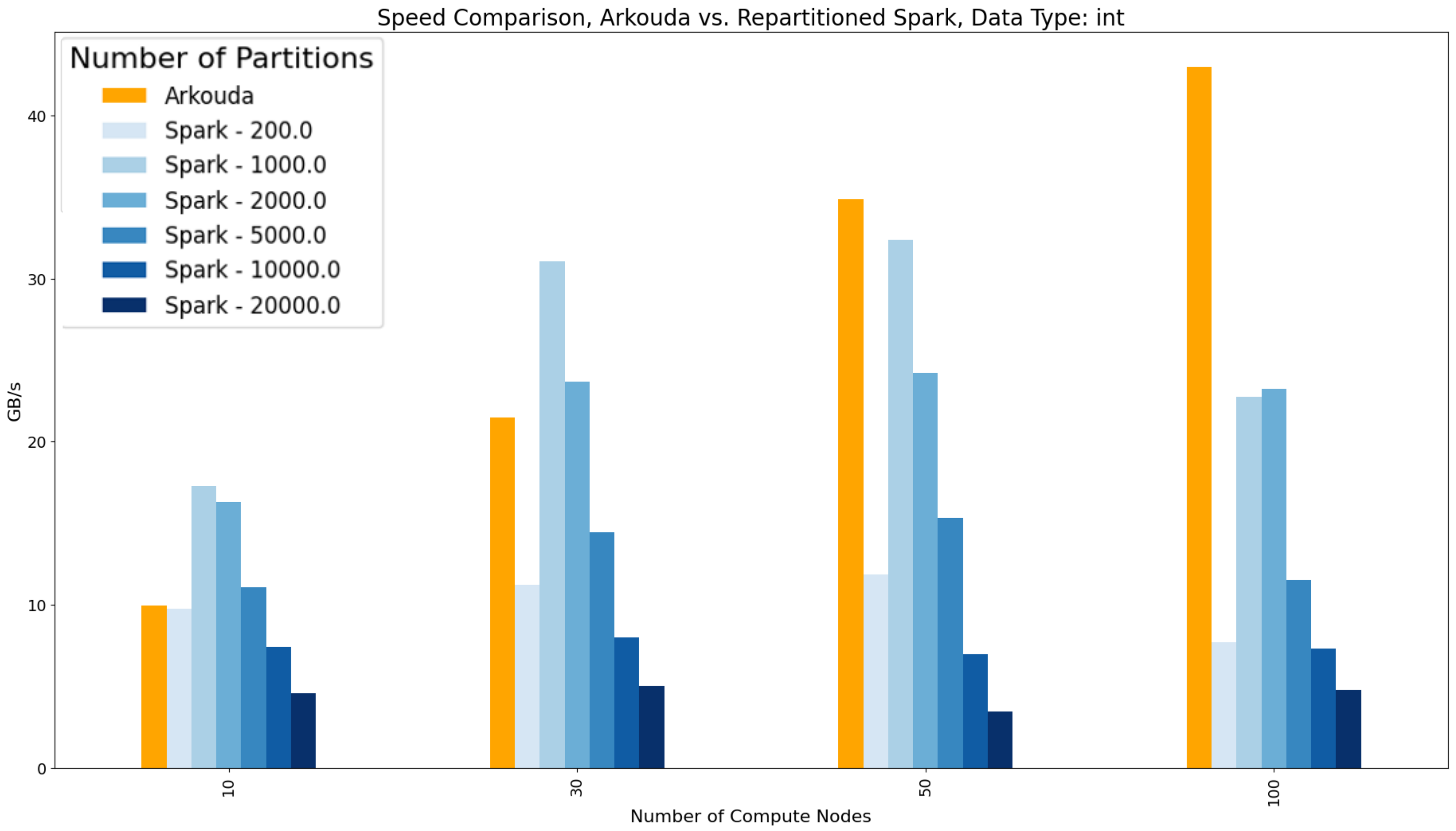
- Boilerplate required to Add Arkouda functionality with this proposal
  - 6 lines of boilerplate code, each is intuitive
  - No repetition for types, since it is handled by the server one level up
  - Looks like “regular” Chapel code, no special Arkouda-specific code other than import



```
module ArraySetopsMsg
{
    use Arkouda;

    proc intersect1dMsg(inArr1: [] ?t, inArr2: [] t) throws {
      return intersect1D(inArr1, inArr2);
    }
}
```

Data Science Beyond the Laptop

Parquet read + GroupBy performance vs Spark



Hewlett Packard Enterprise

Thanks for Attending the ChapelCon 2024 Tutorial Day!

ChapelCon June 5, 2024