Sunrise or Sunset: Exploring the Design Space of Big Data Software Stacks

HPBDC 2017 3rd IEEE International Workshop on High-Performance Big Data Computing

May 29, 2017

gcf@indiana.edu

http://www.dsc.soic.indiana.edu/, http://spidal.org/ http://hpc-abds.org/kaleidoscope/

Department of Intelligent Systems Engineering

School of Informatics and Computing, Digital Science Center

Indiana University Bloomington

Panel Tasking

- The panel will discuss the following three issues:
- Are big data software stacks mature or not?
  - If yes, what is the new technology challenge?
  - If not, what are the main driving forces for new-generation big data software stacks?
  - What chances are provided for the academia communities in exploring the design spaces of big data software stacks?

Are big data software stacks mature or not?

- The solutions are numerous and **powerful**
  - One can get good (and bad) **performance** in an **understandable** reproducible fashion
  - Surely need better documentation and packaging
- The **problems** (and users) are definitely **not mature** (i.e. not understood, and key issues not agreed)
  - Many academic fields are **just starting** to use big data and some are still restricted to small data
  - e.g. **Deep Learning** is not understood in many cases outside the well publicized commercial cases (voice, translation, images)
- In many areas, applications are **pleasingly parallel** or involve MapReduce; performance issues are different from HPC
  - Common is lots of independent Python or R jobs

Why use Spark Hadoop Flink rather than HPC?

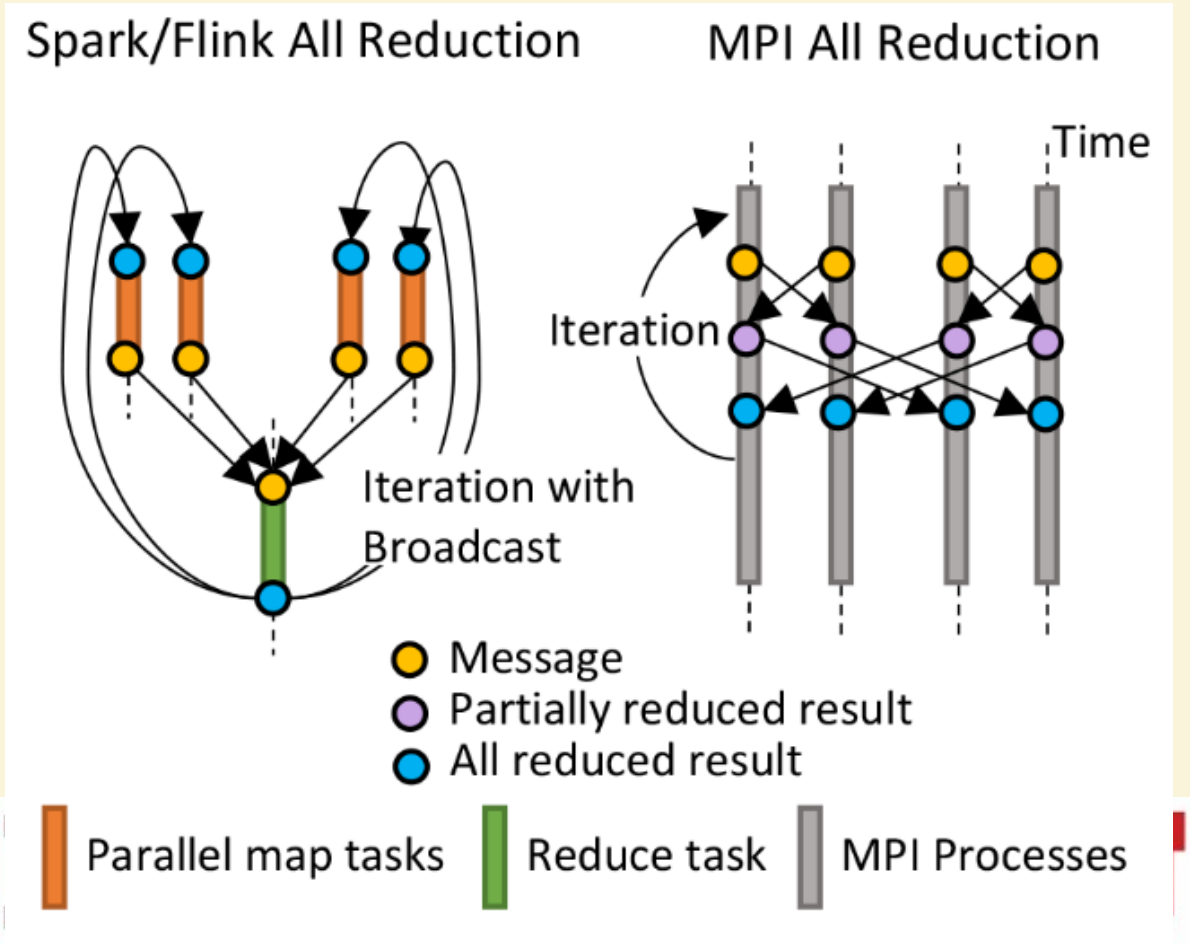
- Yes if you value **ease of programming** over **performance**.
  - This could be the case for most companies where they can find people who can **program in Spark/Hadoop** much more easily than people who can program in MPI.
  - Most of the complications including data, communications are abstracted away to **hide the parallelism** so that average programmer can use Spark/Flink easily and doesn't need to manage state, deal with file systems etc.
  - **RDD data support** very helpful
- For large data problems involving **heterogeneous data sources** such as HDFS with unstructured data, databases such as HBase etc
- Yes if one needs **fault tolerance** for our programs.
  - Our 13-node Moe “big data” (Hadoop twitter analysis) cluster at IU faces such problems around once per month. One can always restart the job, but automatic fault tolerance is convenient.

Why use HPC and not Spark, Flink, Hadoop?

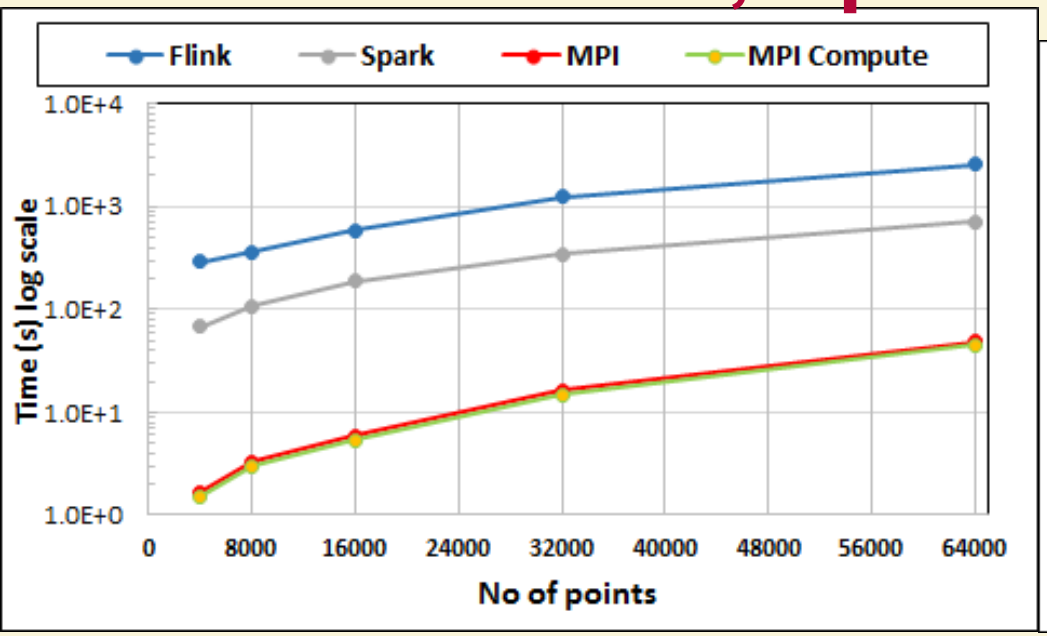
- The performance of Spark, Flink, Hadoop on classic parallel data analytics is **poor/dreadful** whereas HPC (MPI) is **good**
- One way to understand this is to note most Apache systems deliberately support a **dataflow programming model**
- e.g. for Reduce, Apache **will launch a bunch of tasks** and eventually bring results back
  - MPI runs a clever AllReduce interleaved **"in-place" tree**
- Maybe can preserve Spark, Flink programming model but **change implementation** "under the hood" where optimization important.
- Note explicit **dataflow** is efficient and preferred at **coarse scale** as used in workflow systems
  - Need to change implementations for different problems

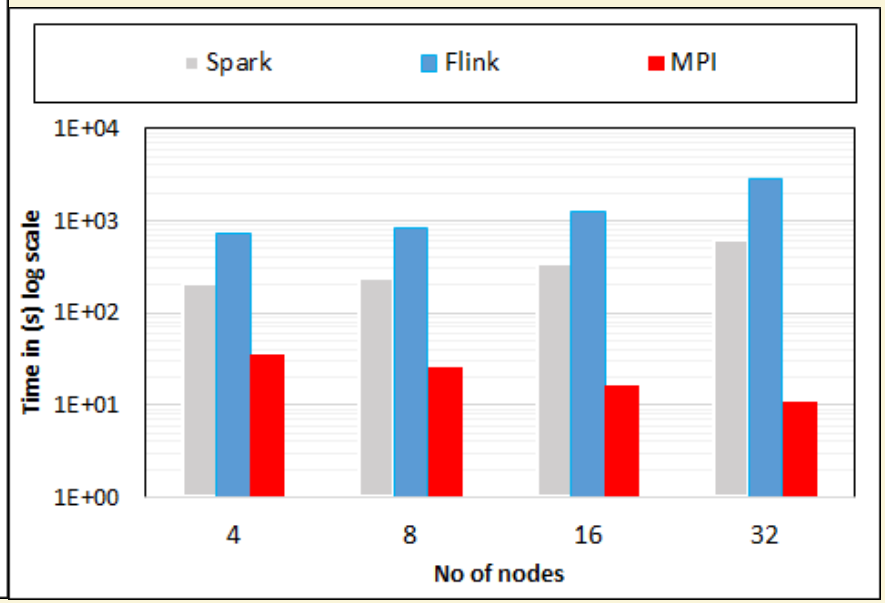
HPC Runtime versus ABDS distributed Computing Model on Data Analytics

Hadoop *writes to disk and is* ***slowest****;* ***Spark*** *and* ***Flink*** *spawn many processes and do not support AllReduce directly;*
MPI *does in-place combined reduce/broadcast and is* ***fastest***



Multidimensional Scaling MDS Results with Flink, Spark and MPI





MDS execution time on 16 nodes with 20 processes in each node with varying number of points

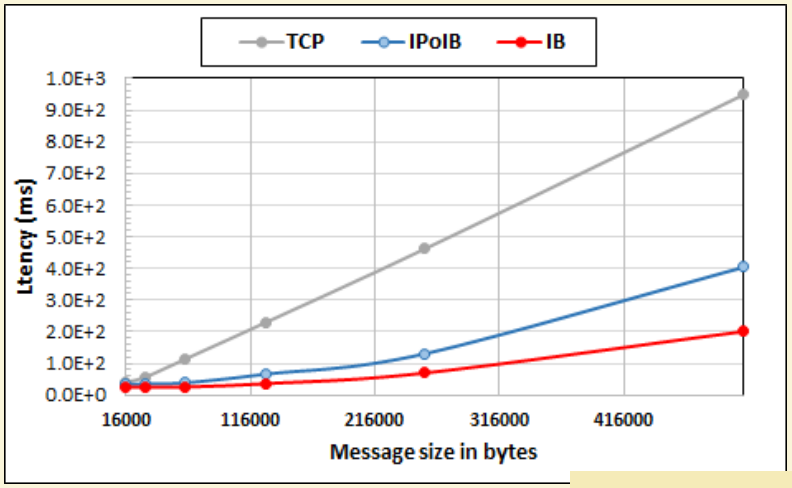
MDS execution time with 32000 points on varying number of nodes. Each node runs 20 parallel tasks

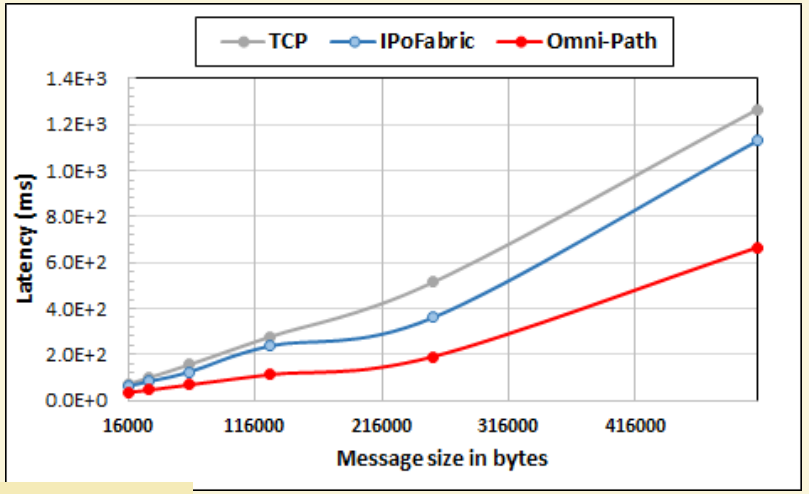
MDS performed poorly on Flink due to its lack of support for nested iterations. In Flink and Spark the algorithm doesn't scale with the number of nodes.

*Intel **Haswell** Cluster with 2.4GHz Processors and 56Gbps Infiniband and 1Gbps Ethernet*

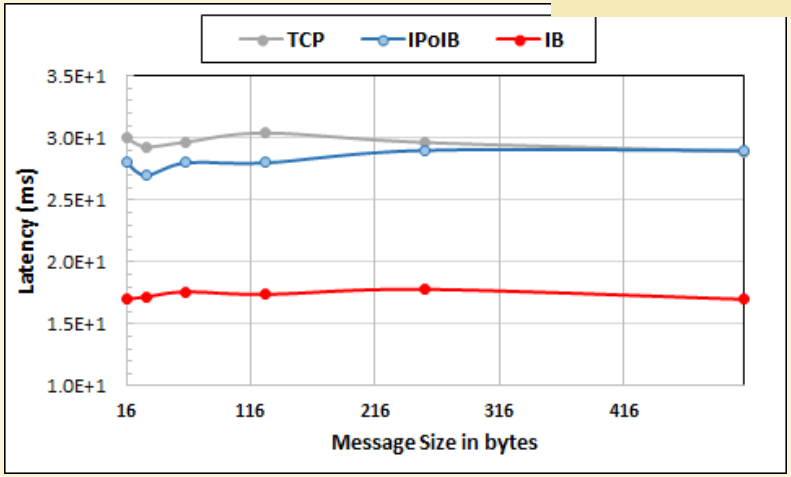
Large messages

*Intel **KNL** Cluster with 1.4GHz Processors and 100Gbps Omni-Path and 1Gbps Ethernet*

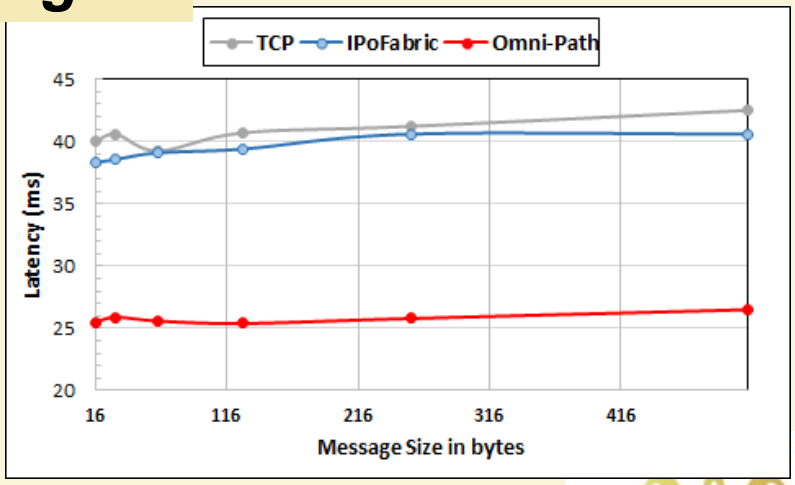




Small messages



Parallelism of 2 and using 8 Nodes

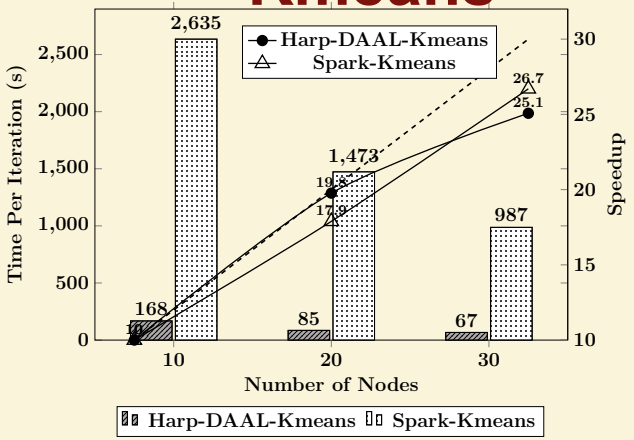


Parallelism of 2 and using 4 Nodes

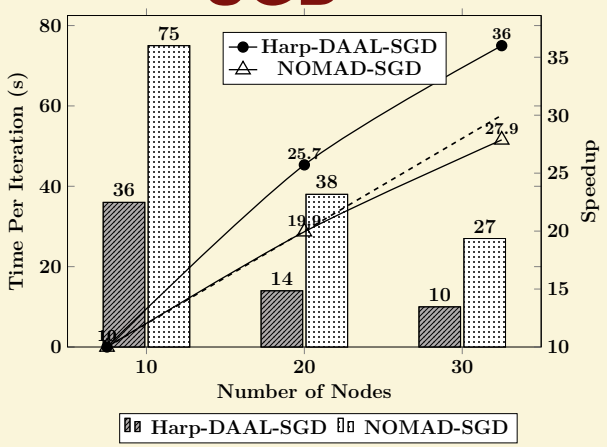
Twitter Heron Streaming Software using HPC Hardware

Knights Landing KNL Data Analytics: Harp, Spark, NOMAD Single Node and Cluster performance: 1.4GHz 68 core nodes

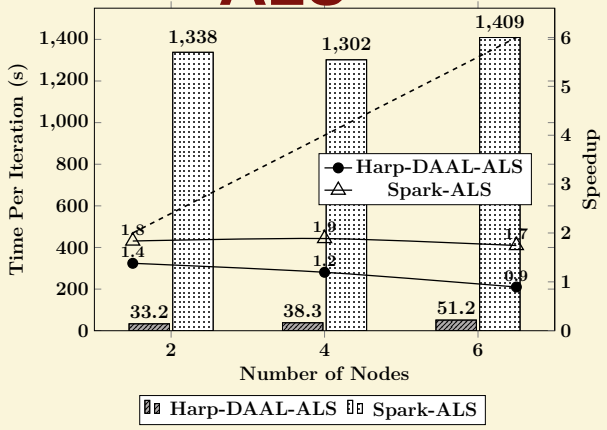
Kmeans



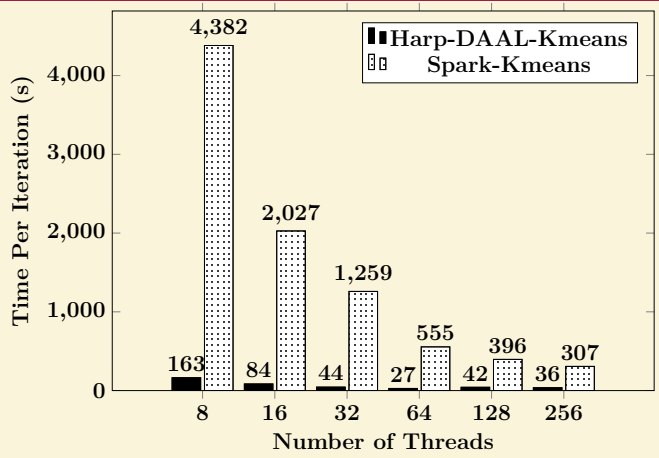
SGD

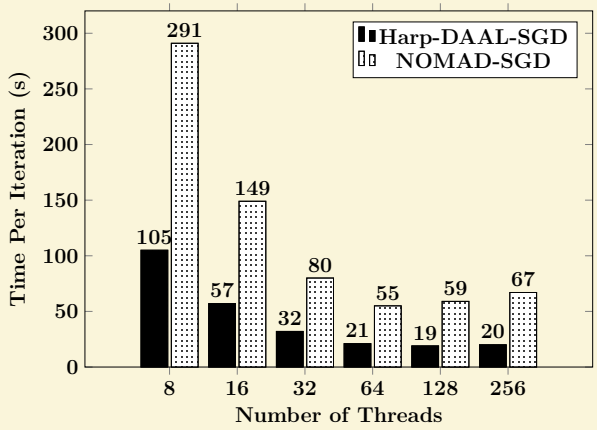


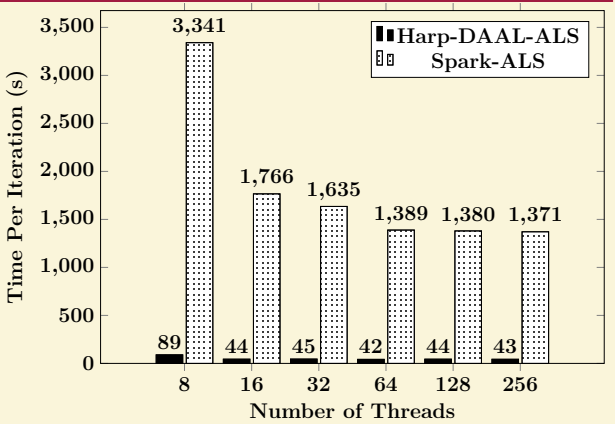
ALS



Strong Scaling Multi Node Parallelism Scaling - Omnipath Interconnect







Strong Scaling Single Node Core Parallelism Scaling

Components of Big Data Stack

- Google likes to show a timeline
- 2002 **Google File System** GFS ~HDFS
- 2004 **MapReduce** Apache Hadoop
- 2006 **Big Table** Apache Hbase
- 2008 **Dremel** Apache Drill
- 2009 **Pregel** Apache Giraph
- 2010 **FlumeJava** Apache Crunch
- 2010 **Colossus** better GFS
- 2012 **Spanner** horizontally scalable NewSQL database ~CockroachDB
- 2013 **F1** horizontally scalable SQL database
- 2013 **MillWheel** ~Apache Storm, Twitter Heron
- 2015 **Cloud Dataflow** Apache Beam with Spark or Flink (dataflow) engine
- Functionalities not identified: **Security, Data Transfer, Scheduling, serverless computing**

What is the new technology challenge?

- **Integrate systems** that offer full capabilities
  - Scheduling
  - Storage
  - "Database"
  - Programming Model (dataflow and/or "in-place" control-flow) and corresponding runtime
  - Analytics
  - Workflow
  - Function as a Service and Event-based Programming

- For both **Batch** and **Streaming**
- **Distributed** and **Centralized** (**Grid** versus **Cluster**)
- Pleasingly parallel (**Local machine learning**) and **Global machine learning** (large scale parallel codes)

What are the main driving forces for new-generation big data software stacks?

- **Applications** ought to **drive** new-generation big data software stacks but (at many universities) academic applications **lag commercial** use in big data area and needs are quite modest
  - This will change and we can expect big data software stacks to become more important
- Note **University** compute systems historically offer **HPC** and not Big Data Expertise.
  - We could anticipate users moving to **public clouds** (away from university systems) but
  - **Users will still want support**
- Need a **Requirements Analysis** that builds in application changes that might occur as users get more sophisticated
- Need to help (**train**) **users** to explore big data opportunities

What chances are provided for the academia communities in exploring the design spaces of big data software stacks?

- We need **more sophisticated applications** to probe some of the most interesting areas
- But most near term opportunities are in **pleasing parallel** (often streaming data) areas
  - Popular technology like **DASK** http://dask.pydata.org for parallel NumPy is pretty inefficient
- Clarify when to use **dataflow** (and Grid technology) and
- When to use **HPC parallel computing** (MPI)
- Likely to require careful consideration of **grain size** and **data distribution**
- Certain to involve **multiple mechanisms** (hidden from user) if want highest performance (combine HPC and Apache Big Data Stack)