Explore Spark for Metagenome assembly

Zhong Wang, Ph.D.
Group Lead, DOE Joint Genome Institute
Lawrence Berkeley National Lab

SPARK SUMMIT EAST 2017

Metagenome is the genome of a microbial community

Microbial communities are "dark matters"

Number of Species

Human
~1000

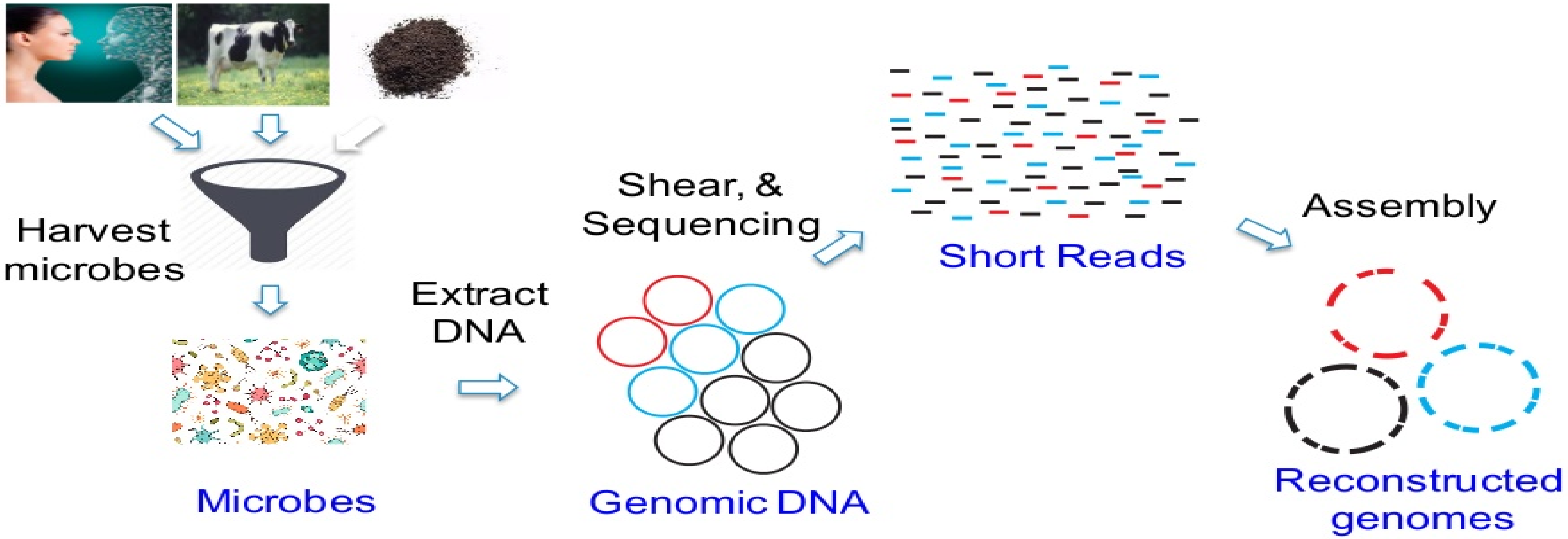
Cow
~6000

Soil,
>100000

>90% of the species haven't been seen before

SPARK SUMMIT EAST 2017

Metagenome sequencing



Metagenome assembly

Genome ~= Book Metagenome ~= Library

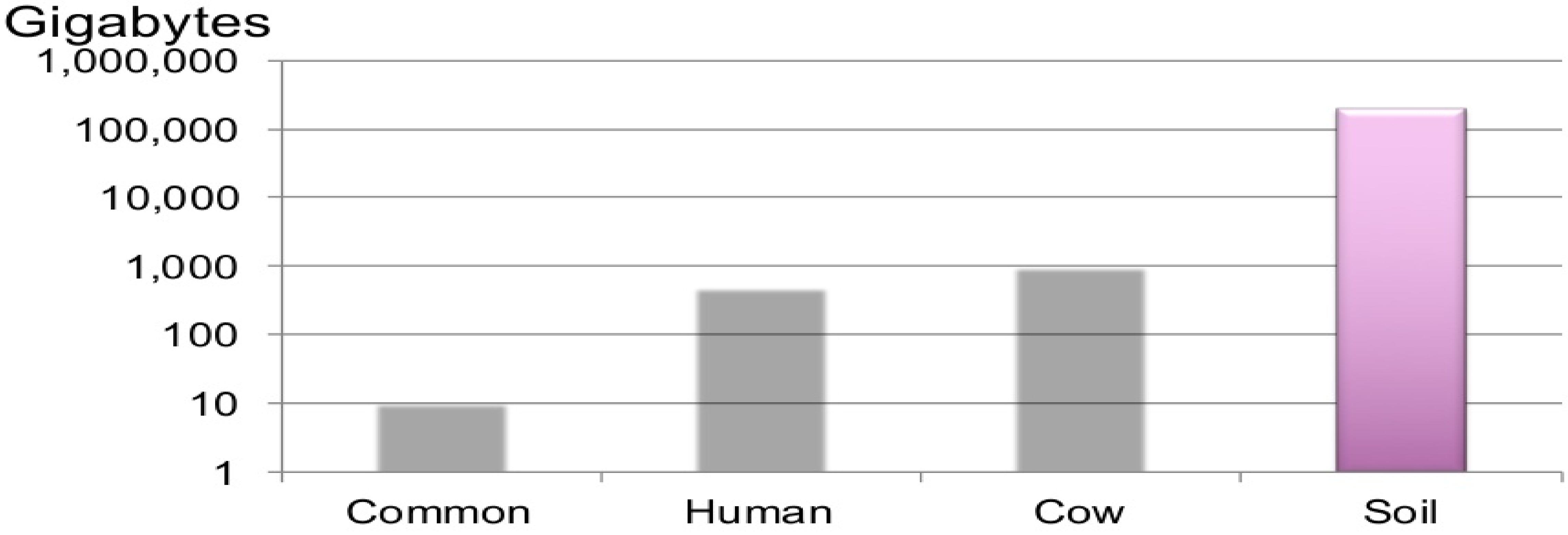
Library of Books

Shredded Library

“reconstructed” Library

Sequencing ~= sampling the pieces

Scale is an enemy



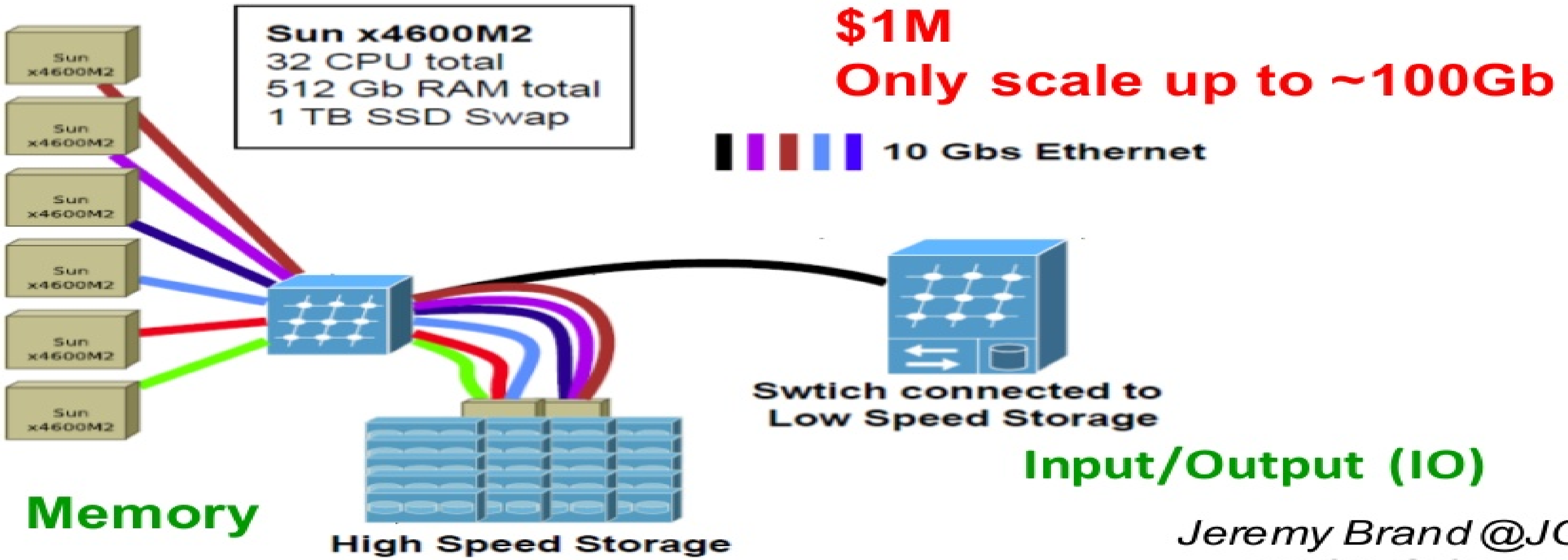
Complexity is another…

- Data complexity
  - Contamination
  - Number of microbial species
  - Species abundance distribution
  - Sequencing errors
- Algorithm complexity
  - Multiple steps, each has different time/space characteristics

The Ideal Solution

- ❑ Easy to develop
- ❑ Robust
- ❑ Scale to big data
- ❑ Efficient

2009: Special Hardware



Jeremy Brand @JGI
FPGA @Convey

2010: MP/MPI on supercomputers

MPI version
412 Gb, 4.5B reads
2.7 hours on 128x24 cores
NESRC Supercomputer

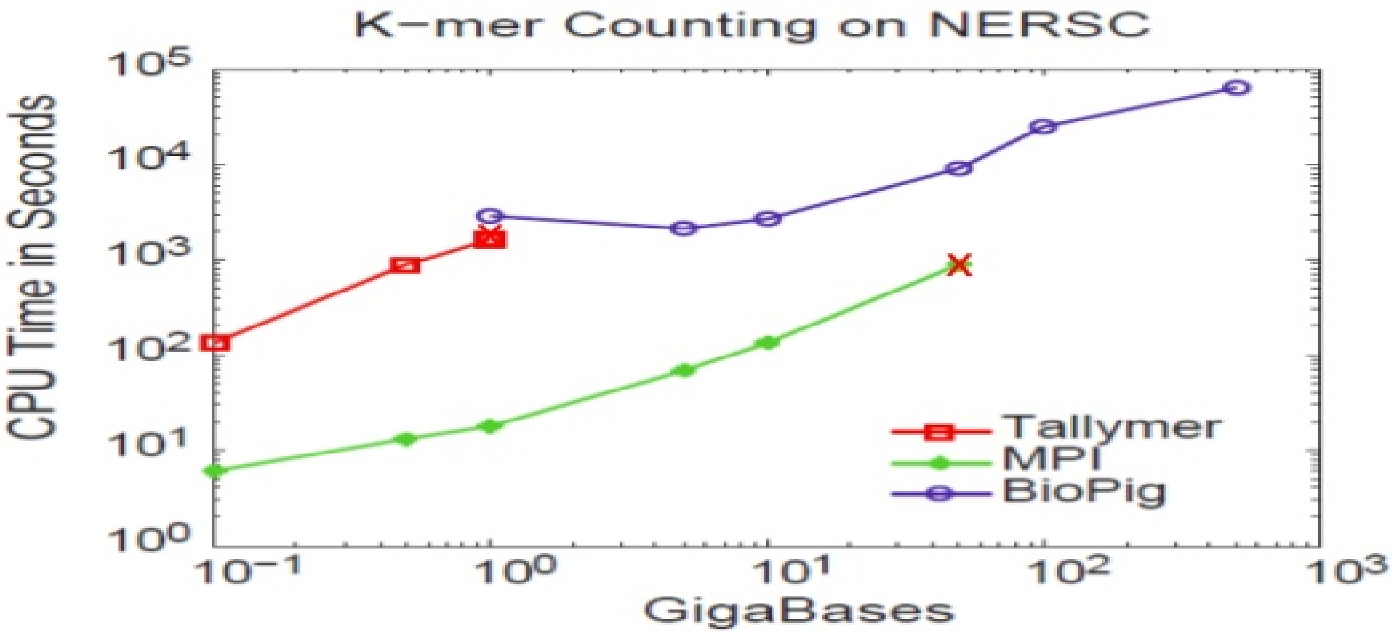
Problems:

Fast, scalable

- Experienced software engineers
- Six months of development time
- One task fails, all tasks fail

Rob Egan @JGI

2013: Hadoop/BioPig



Karan Bhatia, Henrik Nordberg, Kai Wang

Challenges in application

- 2-3 orders of magnitude slower than MPI
- IO optimization, e.g., reduce data copying
- Some problems do not easily fit into map/reduce framework, e.g., graph-based algorithms
- Runs on AWS, but cost $$$ if not optimized

Addressing big data: Apache Spark

- New **scalable** programming paradigm
  - Compatible with Hadoop-supported storage systems
- Improves **efficiency** through:
  - In-memory computing primitives
  - General computation graphs
- Improves **usability** through:
  - Rich APIs in Java, Scala, Python
  - Interactive shell

❑ Scale to big data

❑ Efficient

❑ Easy to develop

❑ Robust?

Goal: Metagenome read clustering

- Clustering reads based on their genome of origin can reduce metagenome problem to single-genome problem
- Ideally scale up to TB data sizes

Algorithm

Contamination prediction

- highly frequent k-mers (n-grams)

K-mer generation and filtering

- K-mer generation
- Filter out low frequency k-mers (noise)

Read graph generation and filtering

- Compute edge weight
- Remove low weight edges

Graph partition

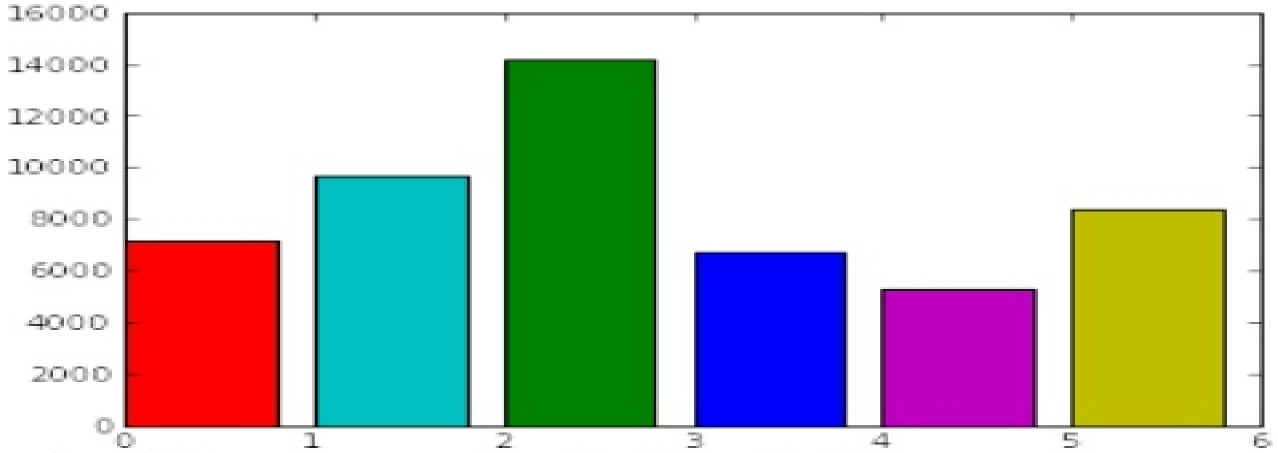
- Connected component
- Power-iteration clustering

Platforms we run spark

- Standalone Spark on single large memory server
- On-demand Spark cluster over HPC
- AWS Elastic Map Reduce (EMR)

Testing the accuracy of the algorithm with a small toy test dataset

- Species:
  - 6 bacterial species (10kb from each)
  - Synthetic communities with random proportions of each genome, reads drawn from single genome sequencing projects (noisy)
  - Ideal situation (no shared sequences between genomes, sufficient sequencing coverage):

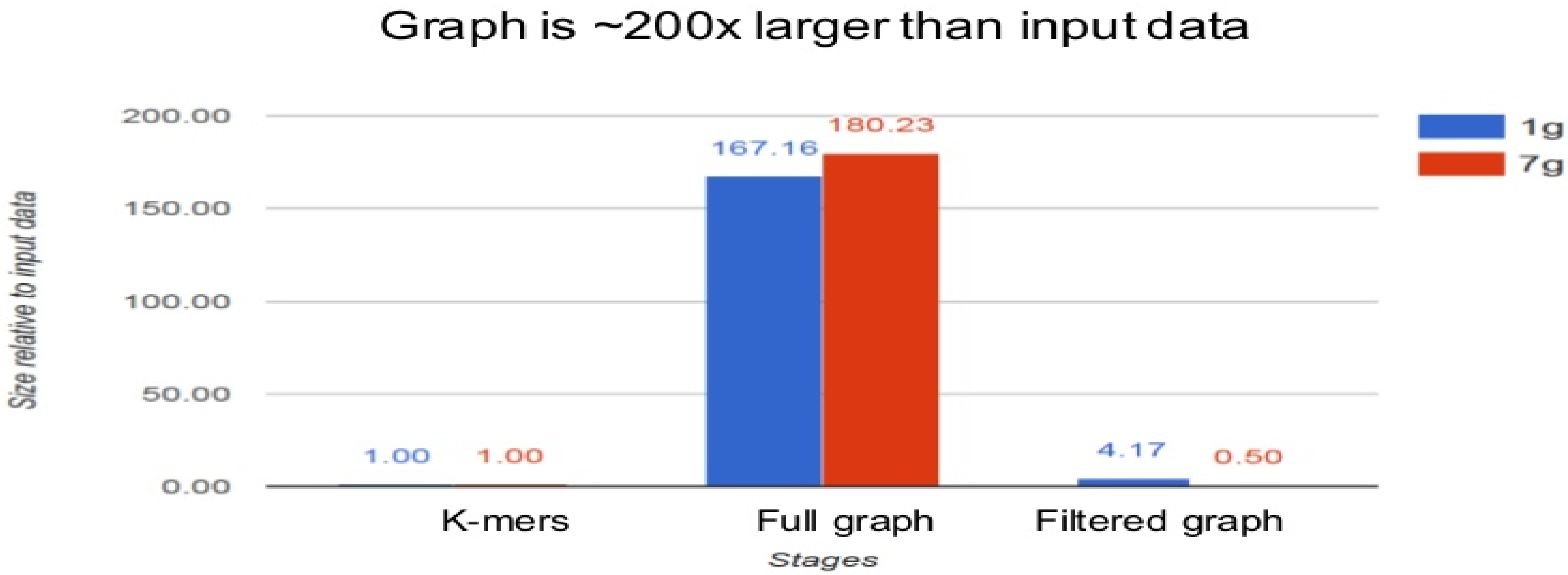


Reads of the same color belong to the same genome

Real world datasets

Dataset	Number of spieces	Sampling depth
Soil metagenome	High	Low
Cow rumen metagenome	Medium	Low to medium
Maize transcriptome (“fake metagenome”)	Low	High

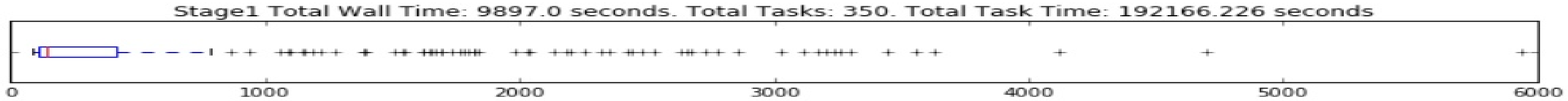
Data grows during analysis



SPARK SUMMIT EAST 2017

Tuning parallelism for load balance

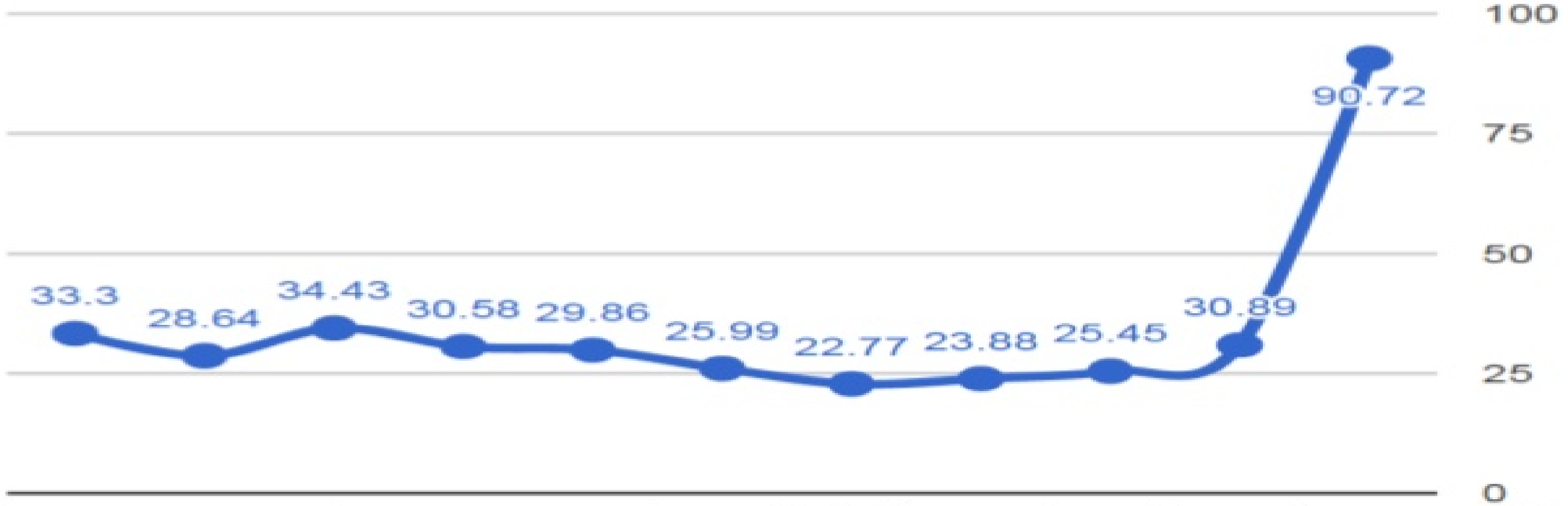
K-mer generation/filtering: partition_size = 16Mb



K-mer generation/filtering: partition_size = 1.3Mb

Tuning parallelism for performance

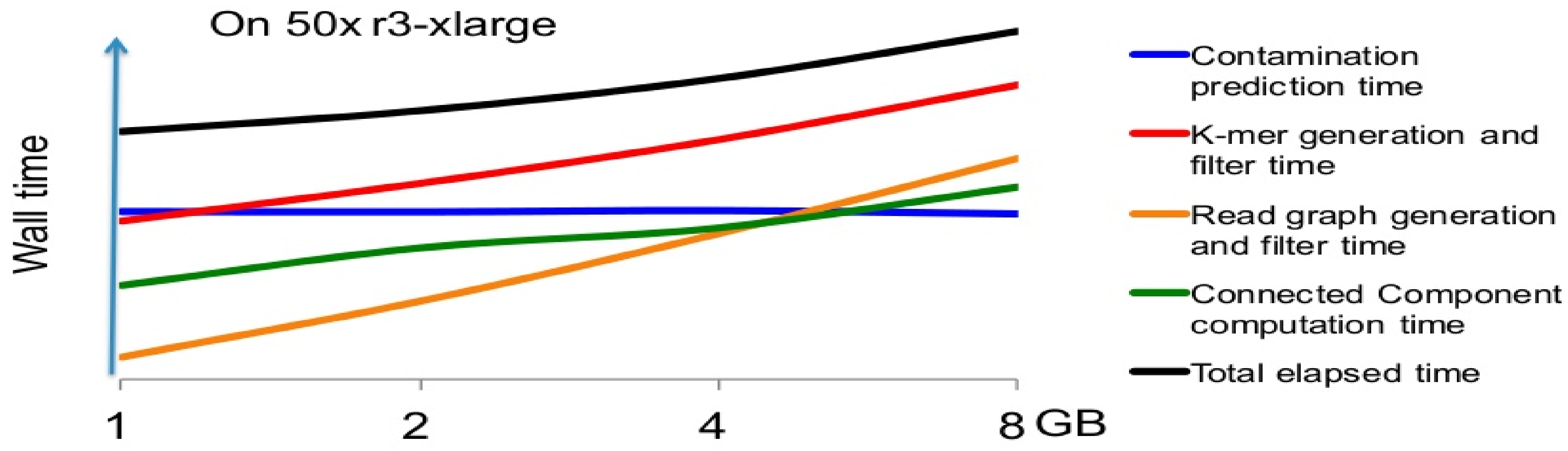
Decreasing partition size for graph construction



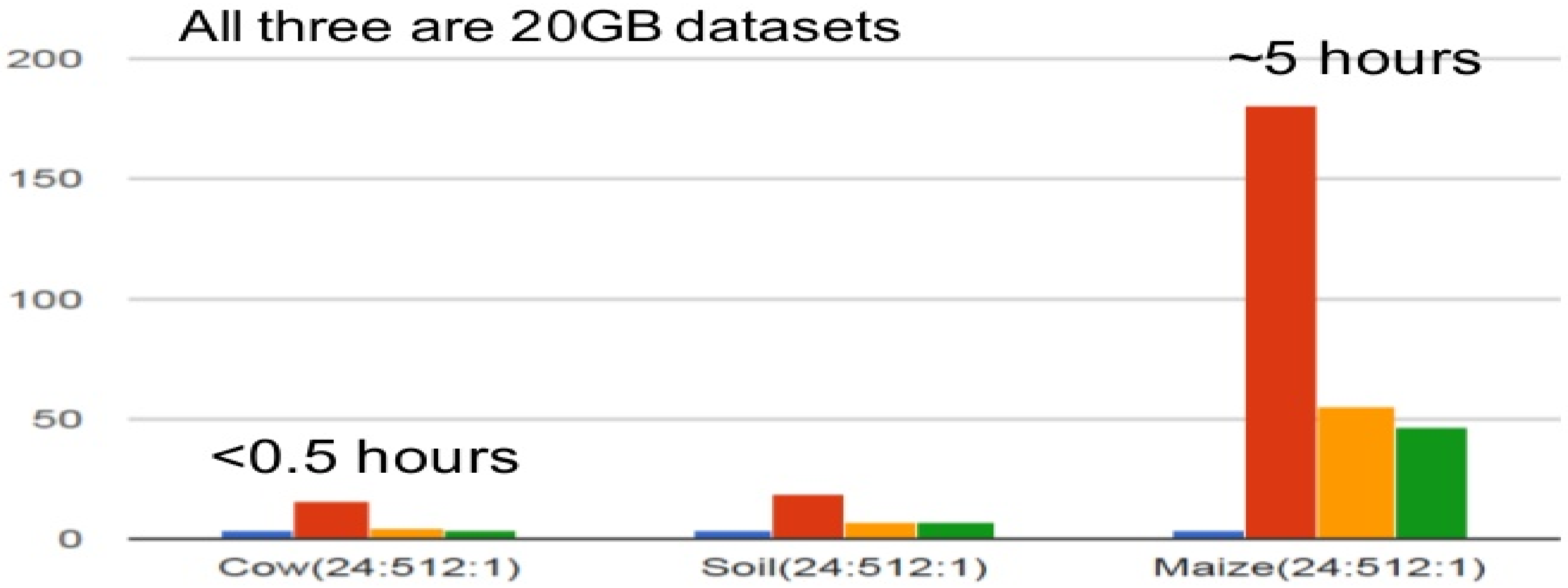
Optimizing parameters can reduce total running time from > 90 min to 20 min

SPARK SUMMIT EAST 2017

Scale well on small data

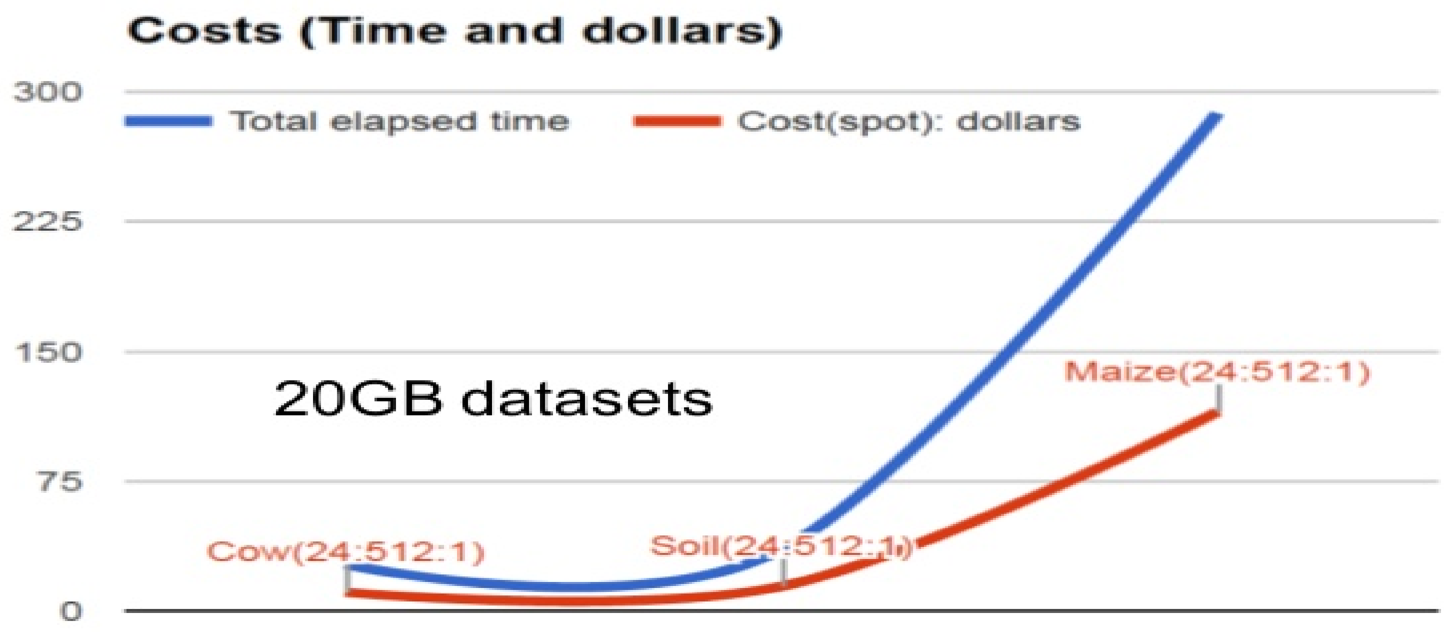


Performance over different datasets



Data complexity is a big driving factor for compute time

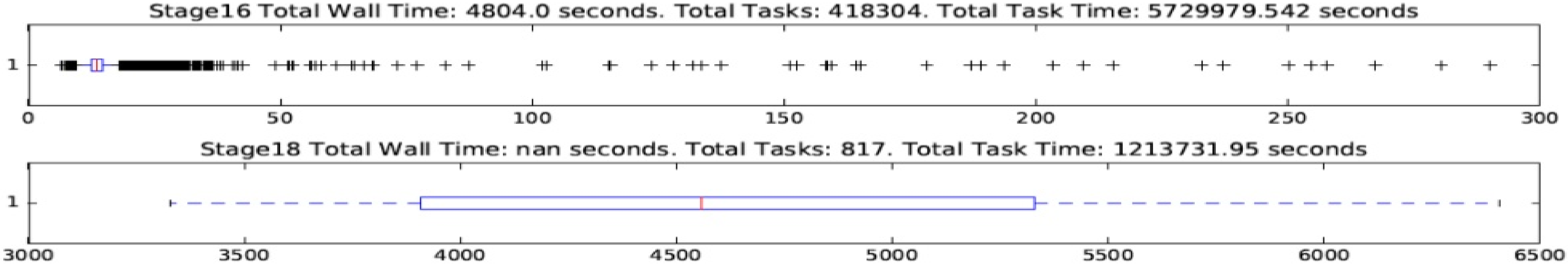
Cost on AWS EMR spot instances



Projected cost for a typical 1TB dataset: ~$500

Scaling up to 100Gb: failed

Read graph generation/filtering



Reducing partition numbers for graph partitioning

SPARK SUMMIT EAST 2017

Potential solutions

- Avoid shuffling:
  - generate the graph, save to disk, then merge partitions outside of Spark.
- Size-specific parameters
  - Larger datasets may not use parallelism parameters optimized for smaller ones
- Your inputs…

Overall impression of Spark

- ✓ **Easy to develop**
  - Scala/python API
  - Databricks notebook
- ✓ **Efficient**
  - Much, much faster than Hadoop
  - High cluster utilization rate

- ? **Robust**
  - Platform dependent
  - 30% failure rate on AWS
- ? **Scale**
  - Problem specific
  - Intermediate data size may change during running
  - Problem complexity may grow with scale

Acknowledgements

Spark Team
Xiandong Meng
Jordan Hoffman@Harvard

JGI

amazon web services™ databricks™

NERSC

Lisa Gerhardt , Evan Racah
@ NERSC

Gary Jung, Greg Kurtzer
Bernard Li,Yong Qin @ HPC

INFORMATION TECHNOLOGY

Thank You.

SPARK SUMMIT EAST 2017