ALLUXIO

EFFECTIVE SPARK WITH ALLUXIO

Jiri Simsa, Alluxio, Inc.

Spark Summit EU

OUTLINE

- Alluxio Overview

- Alluxio + Spark Use Cases

- Using Alluxio with Spark

- Performance Evaluation

ALLUXIO

BIG DATA ECOSYSTEM YESTERDAY
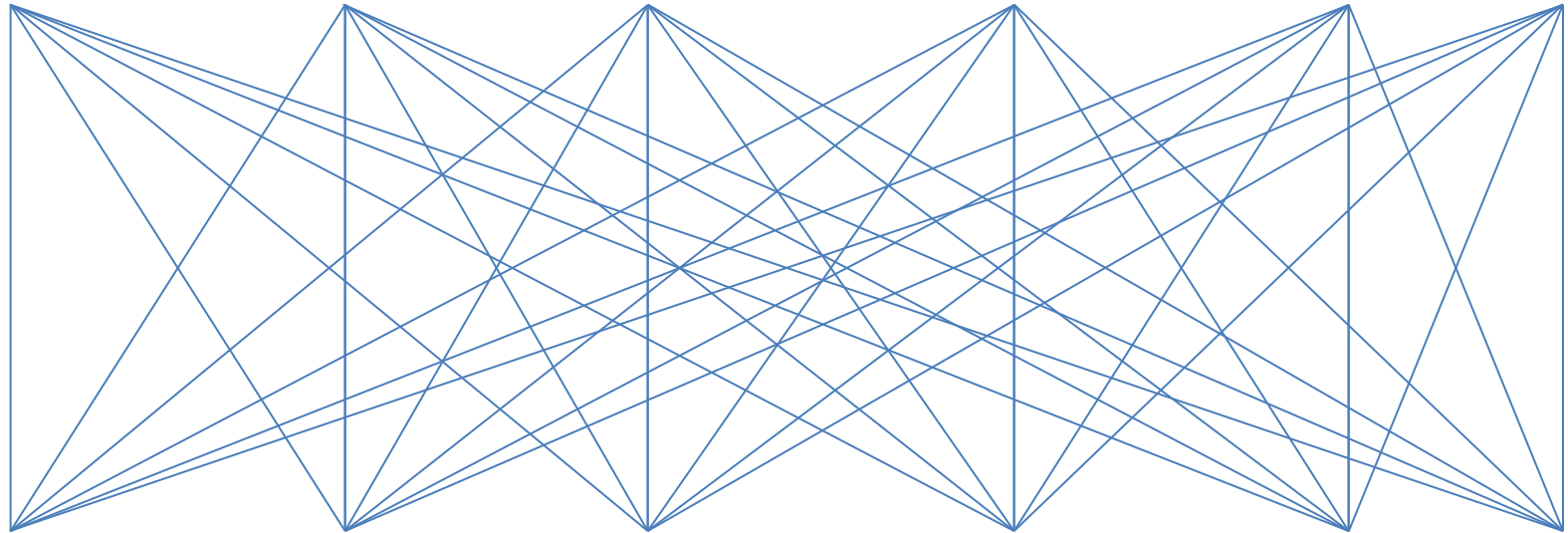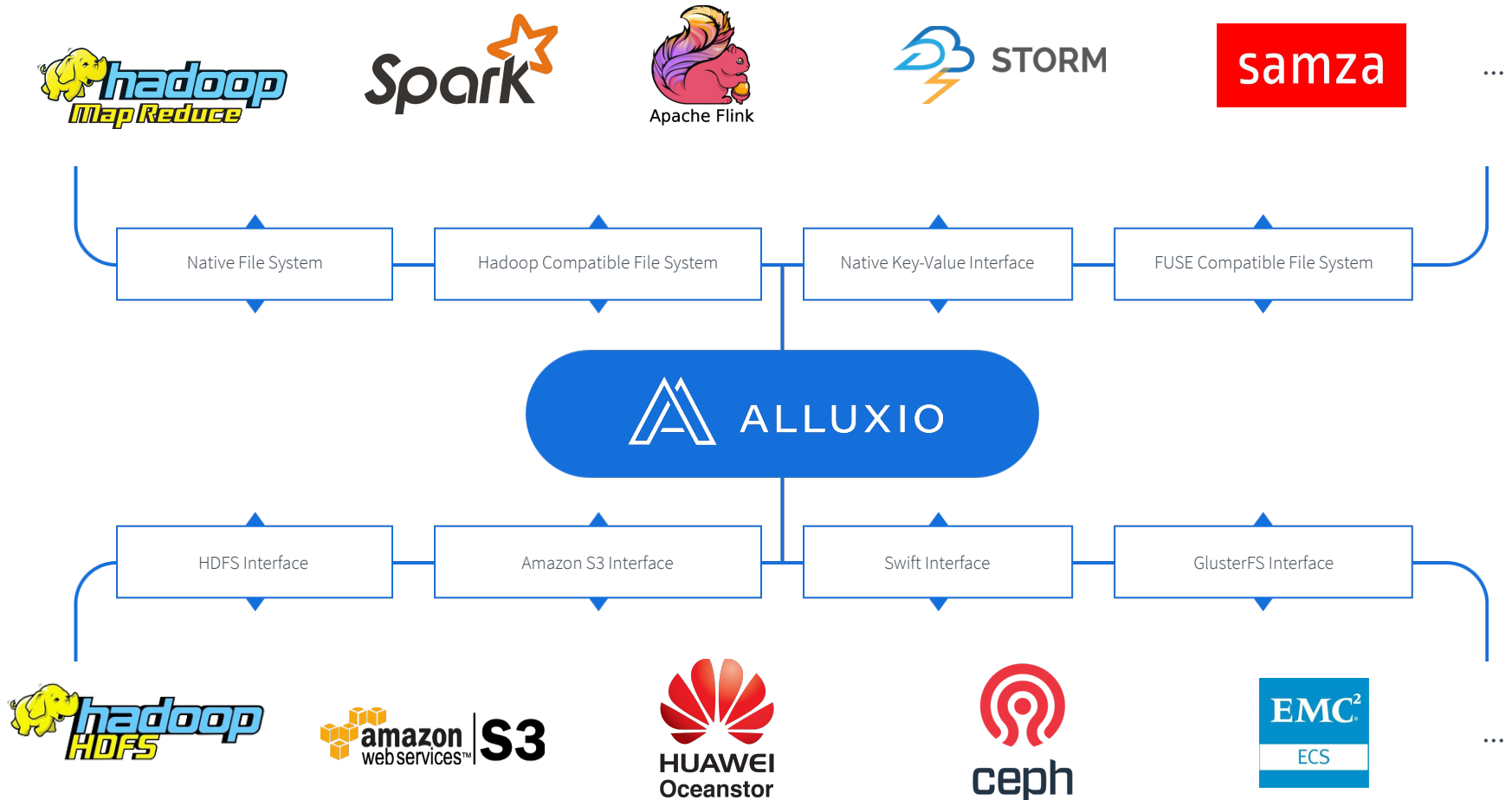
ALLUXIO

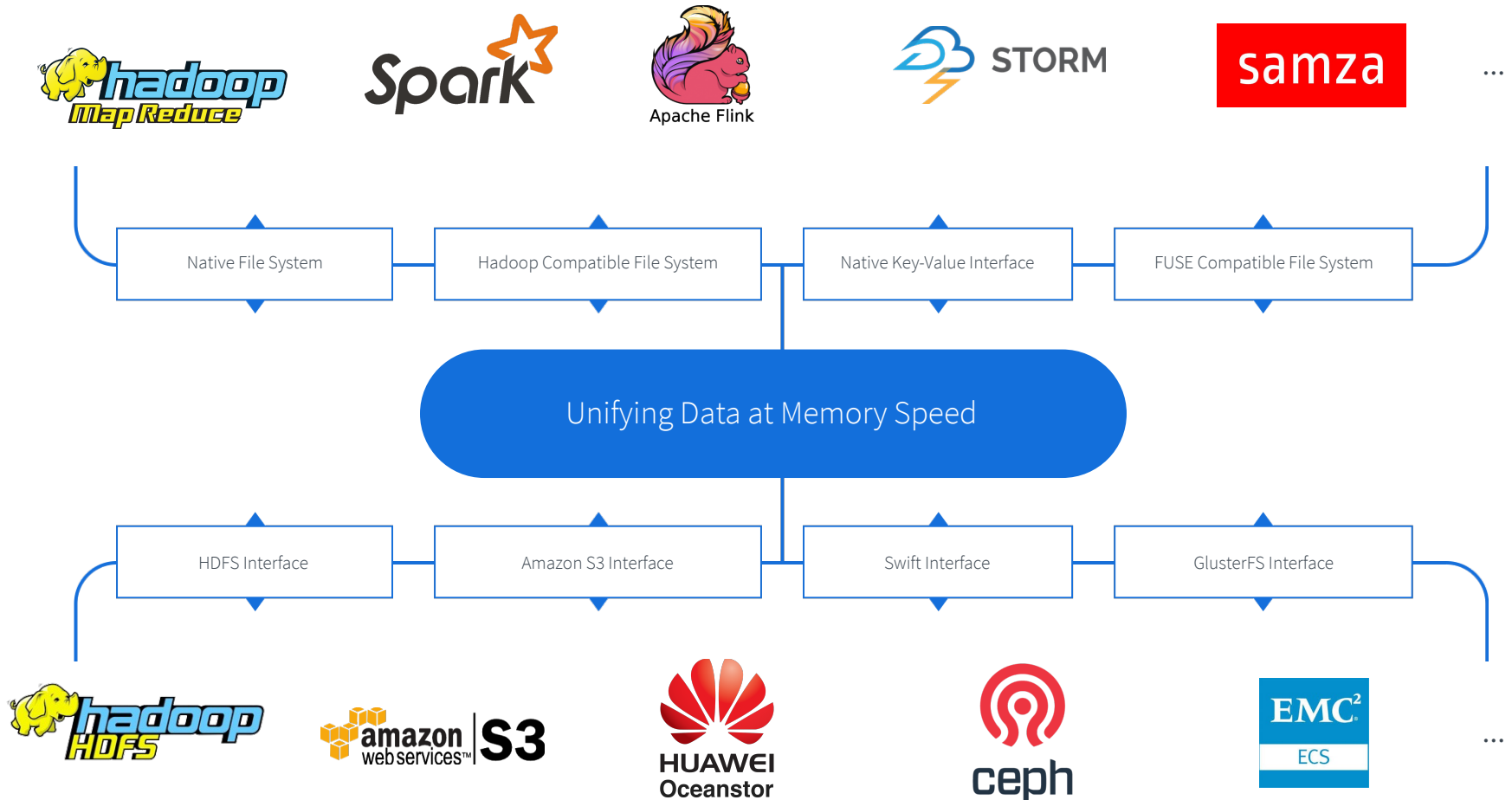# BIG DATA ECOSYSTEM TODAY

ALLUXIO

BIG DATA ECOSYSTEM ISSUES

ALLUXIO

BIG DATA ECOSYSTEM WITH ALLUXIO

BIG DATA ECOSYSTEM WITH ALLUXIO

WHY ALLUXIO

- Co-located compute and data with memory-speed access to data

- Virtualized across different storage systems under a unified namespace

- Scale-out architecture

- File system API, software only

ALLUXIO

BENEFITS

Unification

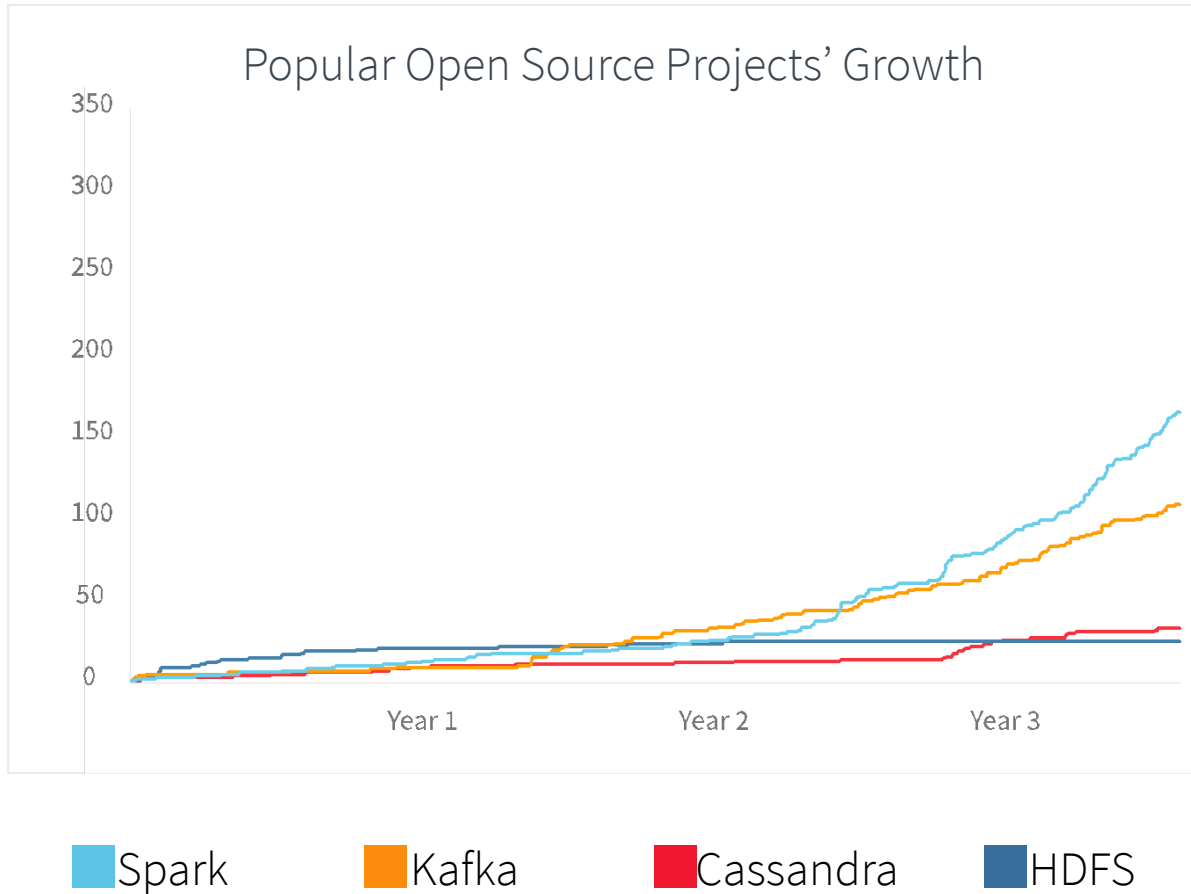New workflows across any data in any storage system

Performance

Orders of magnitude improvement in run time

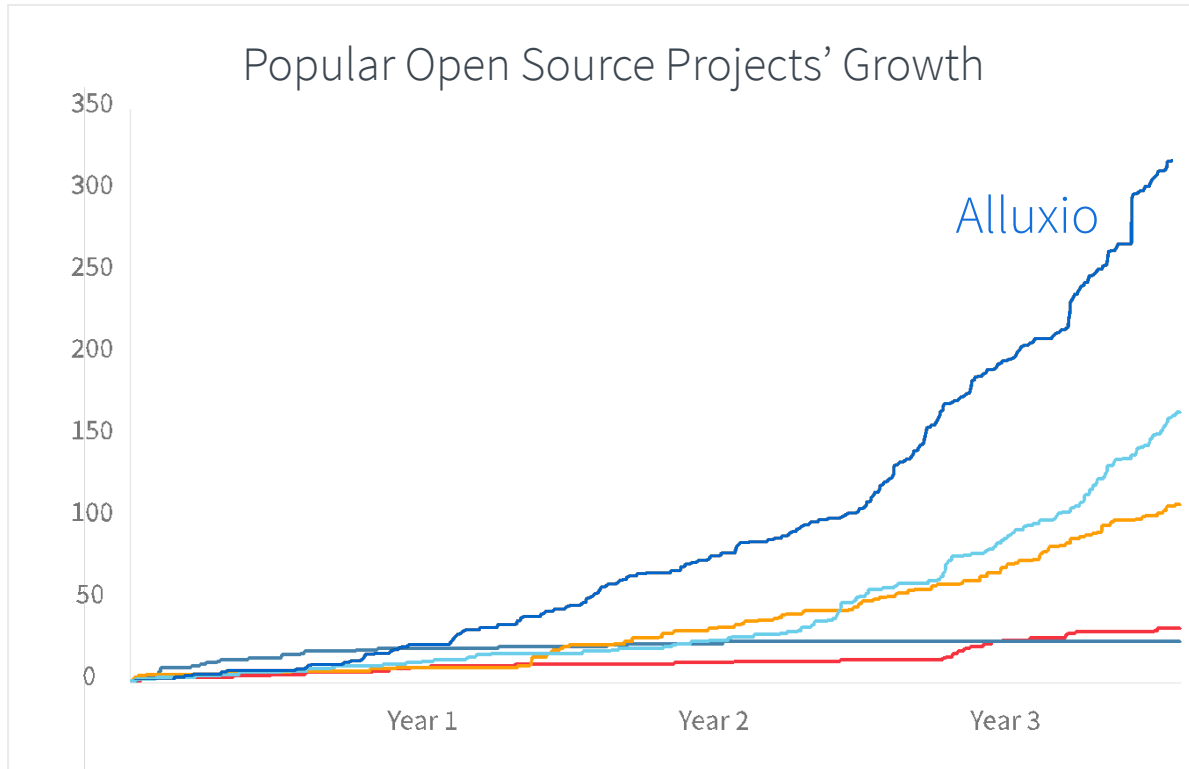### Flexibility

Choice in compute and storage – grow each independently, buy only what is needed

ALLUXIO

FASTEST GROWING BIG DATA PROJECTS

Popular Open Source Projects' Growth

Spark • Kafka • Cassandra • HDFS

ALLUXIO

FASTEST GROWING BIG DATA PROJECTS

Popular Open Source Projects' Growth

- Fastest growing open-source project in the big data ecosystem

- 300+ contributors from 100+ organizations

- Running in large production clusters

Spark   Kafka   Cassandra   HDFS

ALLUXIO

OUTLINE

- Alluxio Overview

- Alluxio + Spark Use Cases

- Using Alluxio with Spark

- Performance Evaluation

ALLUXIO

ACCELERATE I/O TO/FROM REMOTE STORAGE

Baidu's PMs and analysts run

interactive queries to gain insights

into their products and business

Spark

ALLUXIO

Baidu File System

- 200+ nodes deployment
- 2+ petabytes of storage
- Mix of memory + HDD

ALLUXIO

ACCELERATE I/O TO/FROM REMOTE STORAGE

Baidu's PMs and analysts run interactive queries to gain insights into their products and business

Spark

ALLUXIO

Baidu File System

- 200+ nodes deployment
- 2+ petabytes of storage
- Mix of memory + HDD

Bai du 百度

The performance was amazing. With Spark SQL alone, it took 100-150 seconds to finish a query; using Alluxio, where data may hit local or remote Alluxio nodes, it took 10-15 seconds.

- Baidu

RESULTS

- Data queries are now 30x faster with Alluxio

- Alluxio cluster runs stably, providing over 50TB of RAM space

- By using Alluxio, batch queries usually lasting over 15 minutes were transformed into an interactive query taking less than 30 seconds

ALLUXIO

SHARE DATA ACROSS JOBS AT MEMORY SPEED

Barclays uses query and machine learning to train models for risk management

Spark

ALLUXIO

Relational Database

- 6 node deployment
- 1TB of storage
- Memory only

BARCLAYS

ALLUXIO

SHARE DATA ACROSS JOBS AT MEMORY SPEED

Barclays uses query and machine learning to train models for risk management
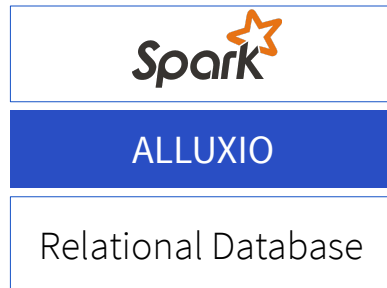
Spark

ALLUXIO

Relational Database

- 6 node deployment
- 1TB of storage
- Memory only

BARCLAYS

❝ *Thanks to Alluxio, we now have the raw data immediately available at every iteration and we can skip the costs of loading in terms of time waiting, network traffic, and RDBMS activity.*

- Barclays

RESULTS

- Barclays workflow iteration time decreased from hours to seconds

- Alluxio enabled workflows that were impossible before

- By keeping data only in memory, the I/O cost of loading and storing in Alluxio is now on the order of seconds

ALLUXIO

MANAGE DATA ACROSS STORAGE SYSTEMS

Qunar uses real-time machine
learning for their website ads.

- 200+ nodes deployment
- 6 billion logs (4.5 TB) daily
- Mix of Memory + HDD

MANAGE DATA ACROSS STORAGE SYSTEMS

Qunar uses real-time machine learning for their website ads.

ALLUXIO

- 200+ nodes deployment
- 6 billion logs (4.5 TB) daily
- Mix of Memory + HDD

We've been running Alluxio in production for over 9 months, Alluxio's unified namespace enable different applications and frameworks to easily interact with data from different storage systems.

- Qunar

RESULTS

- Data sharing among Spark Streaming, Spark batch and Flink jobs provide efficient data sharing

- Improved the performance of their system with 15x – 300x speedups

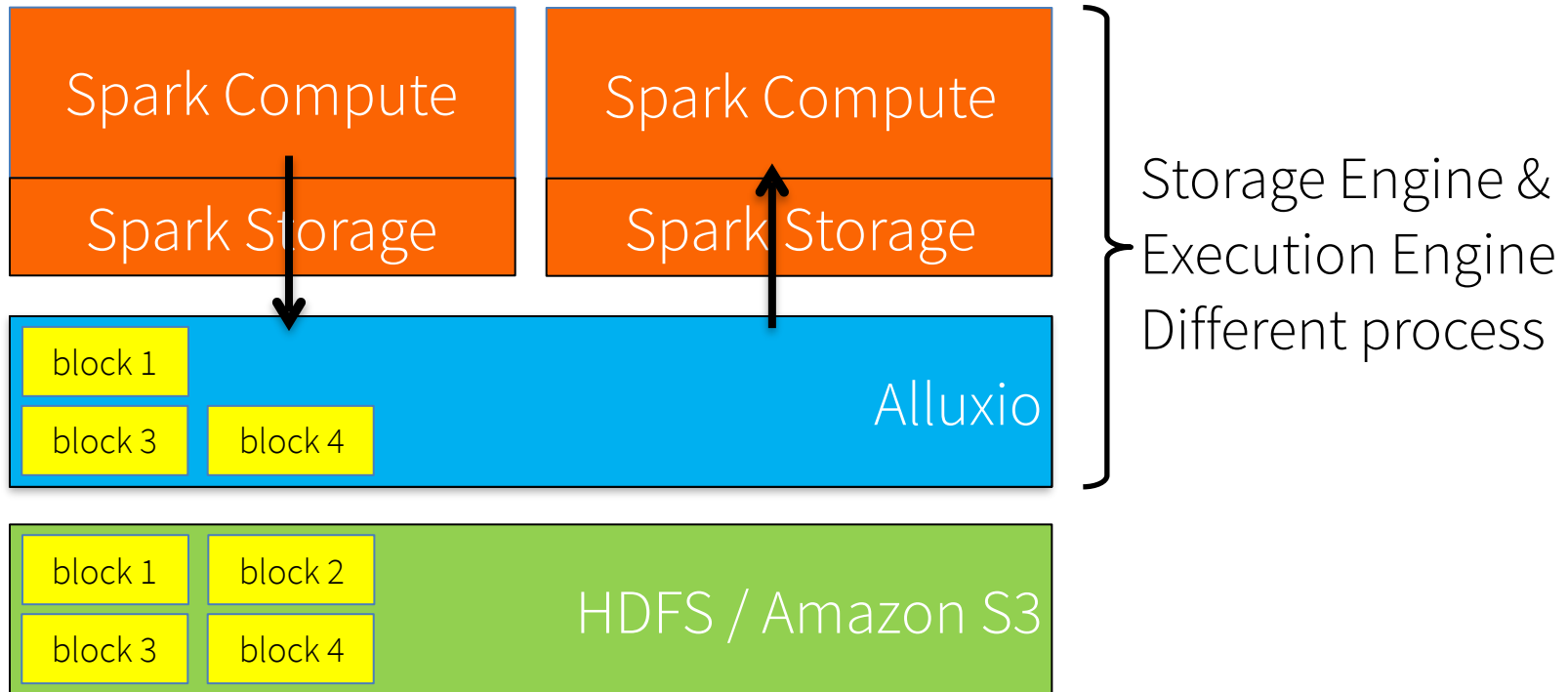- Tiered storage feature manages storage resources including memory and HDD

ALLUXIO

OUTLINE

- Alluxio Overview

- Alluxio + Spark Use Cases

- Using Alluxio with Spark

- Performance Evaluation

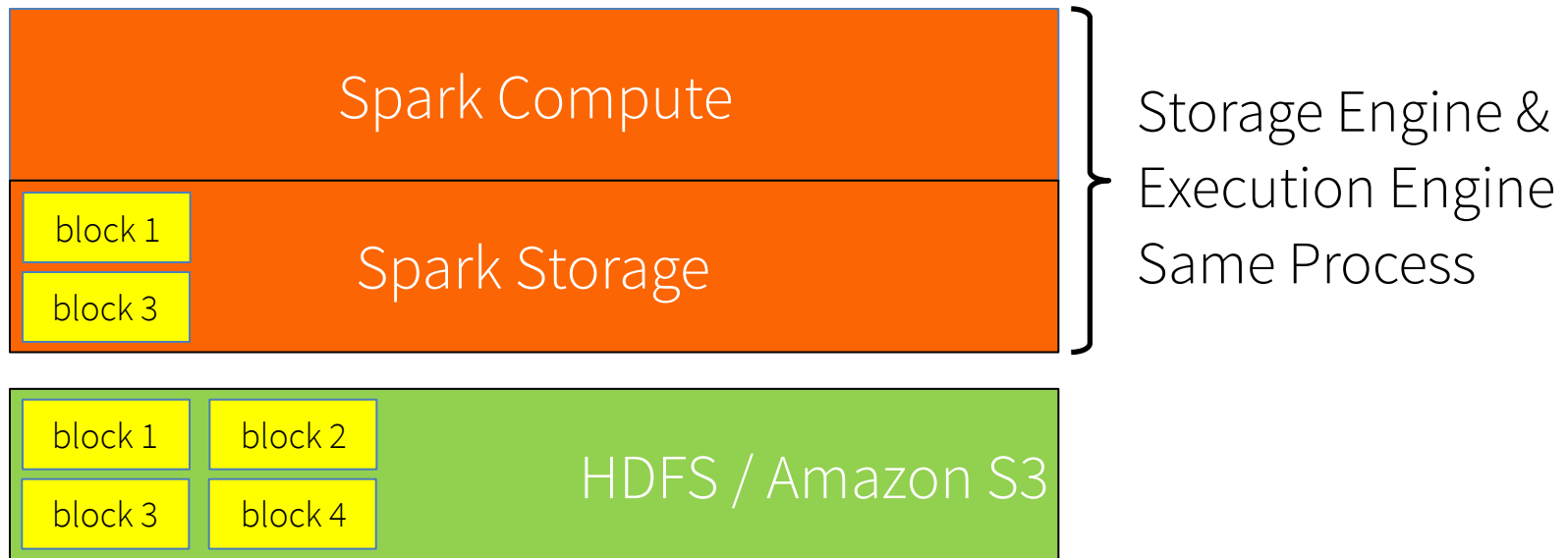ALLUXIO

CONSOLIDATING MEMORY
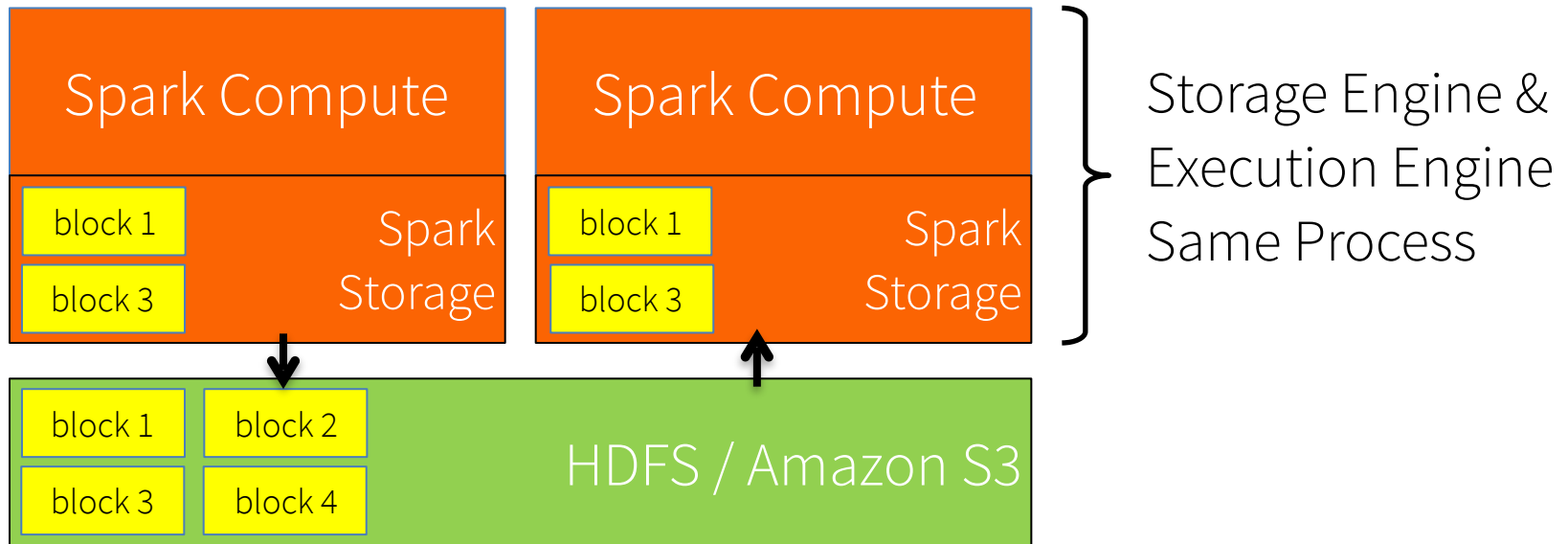
- Two copies of data in memory – double the memory used
- Inter-process Sharing Slowed Down by Network / Disk I/O
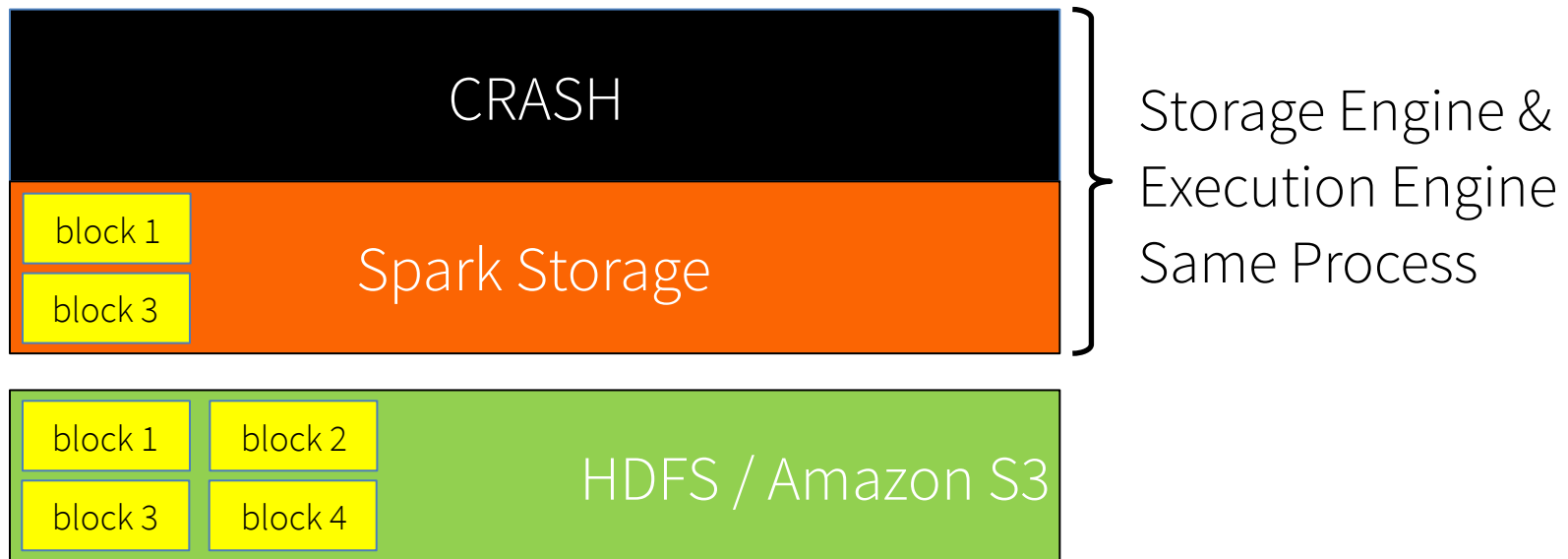
ALLUXIO

CONSOLIDATING MEMORY

- Half the memory used
- Inter-process Sharing Happens at Memory Speed

DATA RESILIENCE DURING CRASH

Spark Compute

Spark Storage

block 1

block 3

HDFS / Amazon S3

block 1

block 2

block 3

block 4
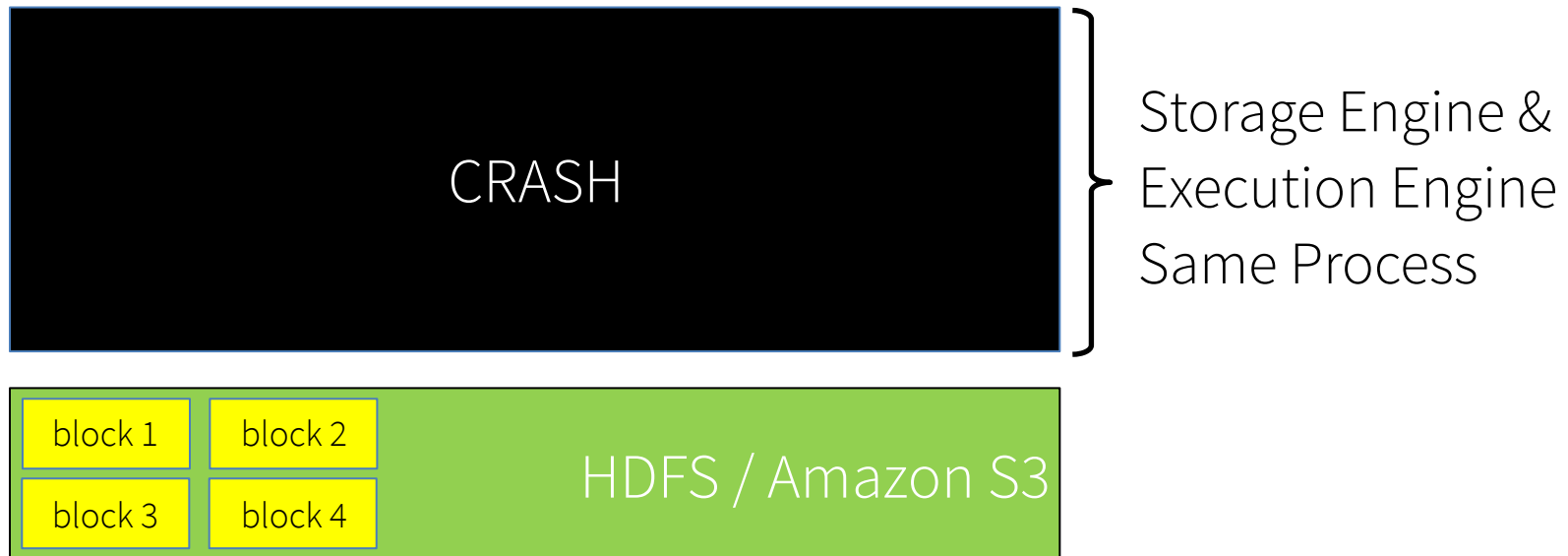
Storage Engine &
Execution Engine
Same Process

ALLUXIO

DATA RESILIENCE DURING CRASH

- Process Crash Requires Network and/or Disk I/O to Re-read Data

ALLUXIO

DATA RESILIENCE DURING CRASH

CRASH

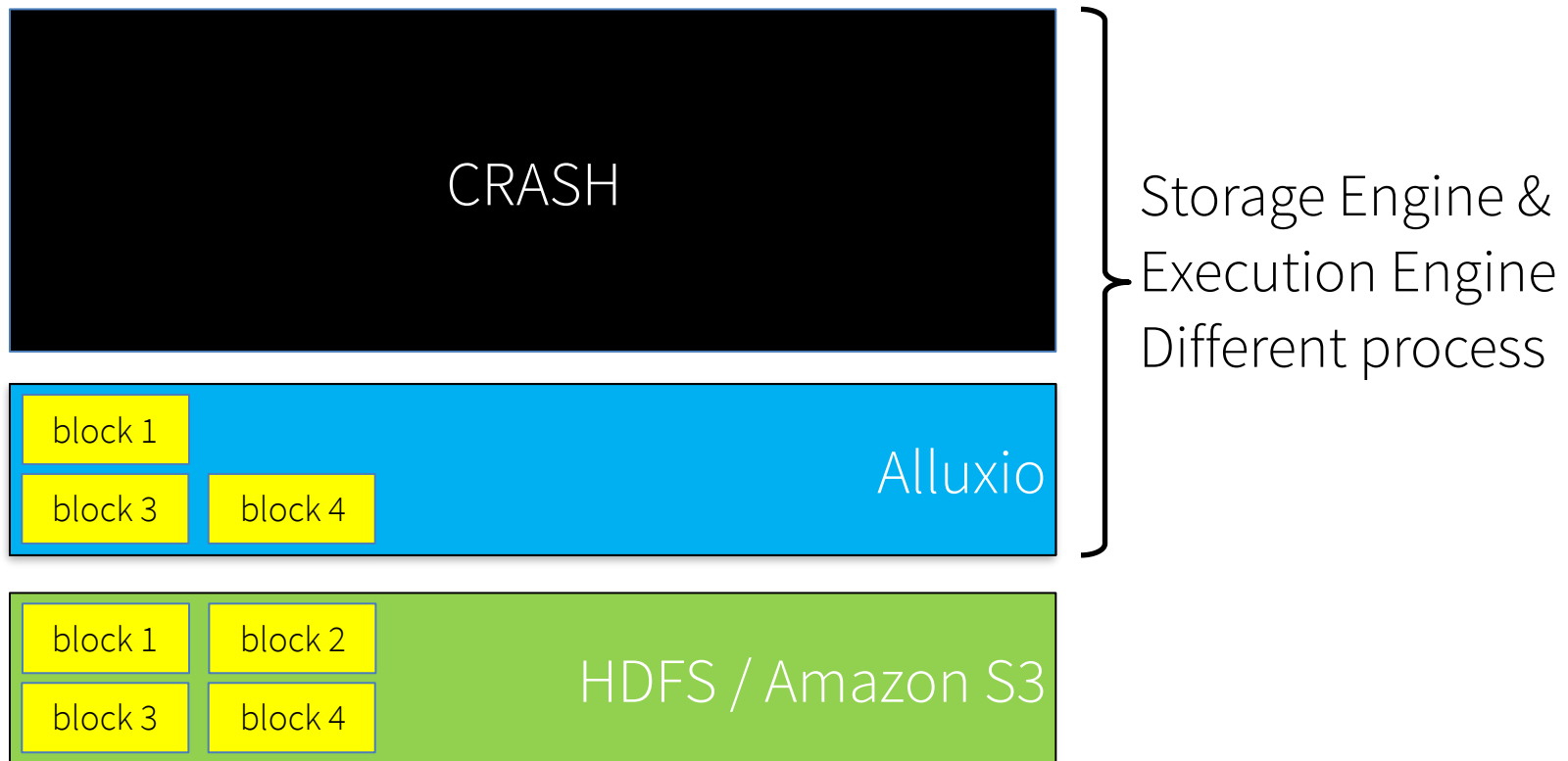Storage Engine & Execution Engine Same Process

block 1   block 2
block 3   block 4

HDFS / Amazon S3

- Process Crash Requires Network and/or Disk I/O to Re-read Data

ALLUXIO

DATA RESILIENCE DURING CRASH

ALLUXIO

DATA RESILIENCE DURING CRASH

CRASH

Storage Engine & Execution Engine Different process

block 1

block 3   block 4

Alluxio

block 1   block 2

block 3   block 4

HDFS / Amazon S3

- Process Crash – Data is Re-read at Memory Speed

ALLUXIO

ACCESSING ALLUXIO DATA FROM SPARK

Writing Data ▶ Write to an Alluxio file

Reading Data ▶ Read from an Alluxio file

ALLUXIO

CODE EXAMPLE FOR SPARK RDDS

Writing RDD
to Alluxio

▶

```
rdd.saveAsTextFile(alluxioPath)
rdd.saveAsObjectFile(alluxioPath)
```

Reading RDD
from Alluxio

▶

```
rdd = sc.textFile(alluxioPath)
rdd = sc.objectFile(alluxioPath)
```

ALLUXIO

CODE EXAMPLE FOR SPARK DATAFRAMES

Writing DataFrame
to Alluxio

▶ `df.write.parquet(alluxioPath)`

Reading DataFrame
from Alluxio

▶ `df = sc.read.parquet(alluxioPath)`

△ ALLUXIO

OUTLINE

- Alluxio Overview

- Alluxio + Spark Use Cases

- Using Alluxio with Spark
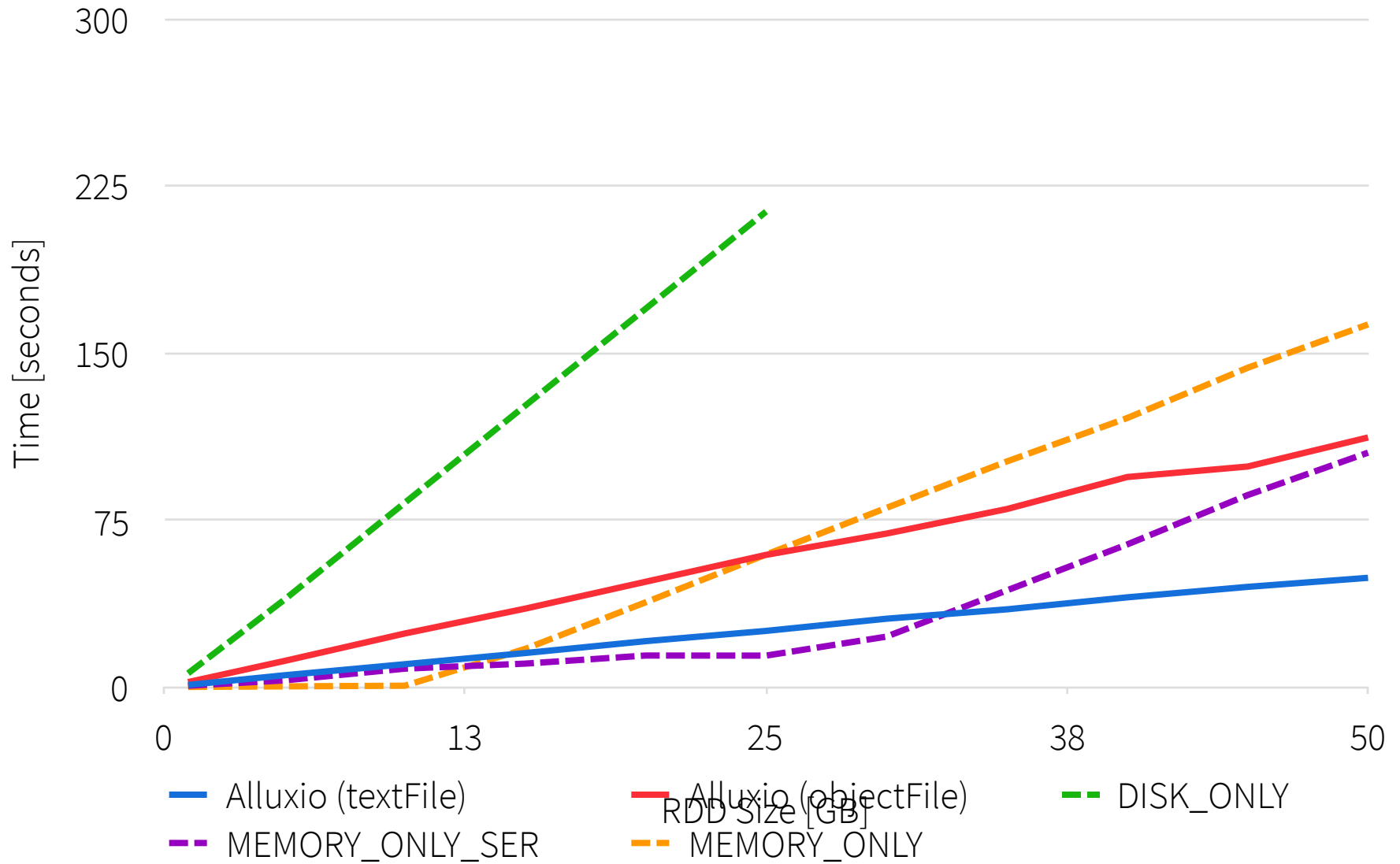
- Performance Evaluation

ALLUXIO

ENVIRONMENT

Spark 2.0.0 + Alluxio 1.2.0
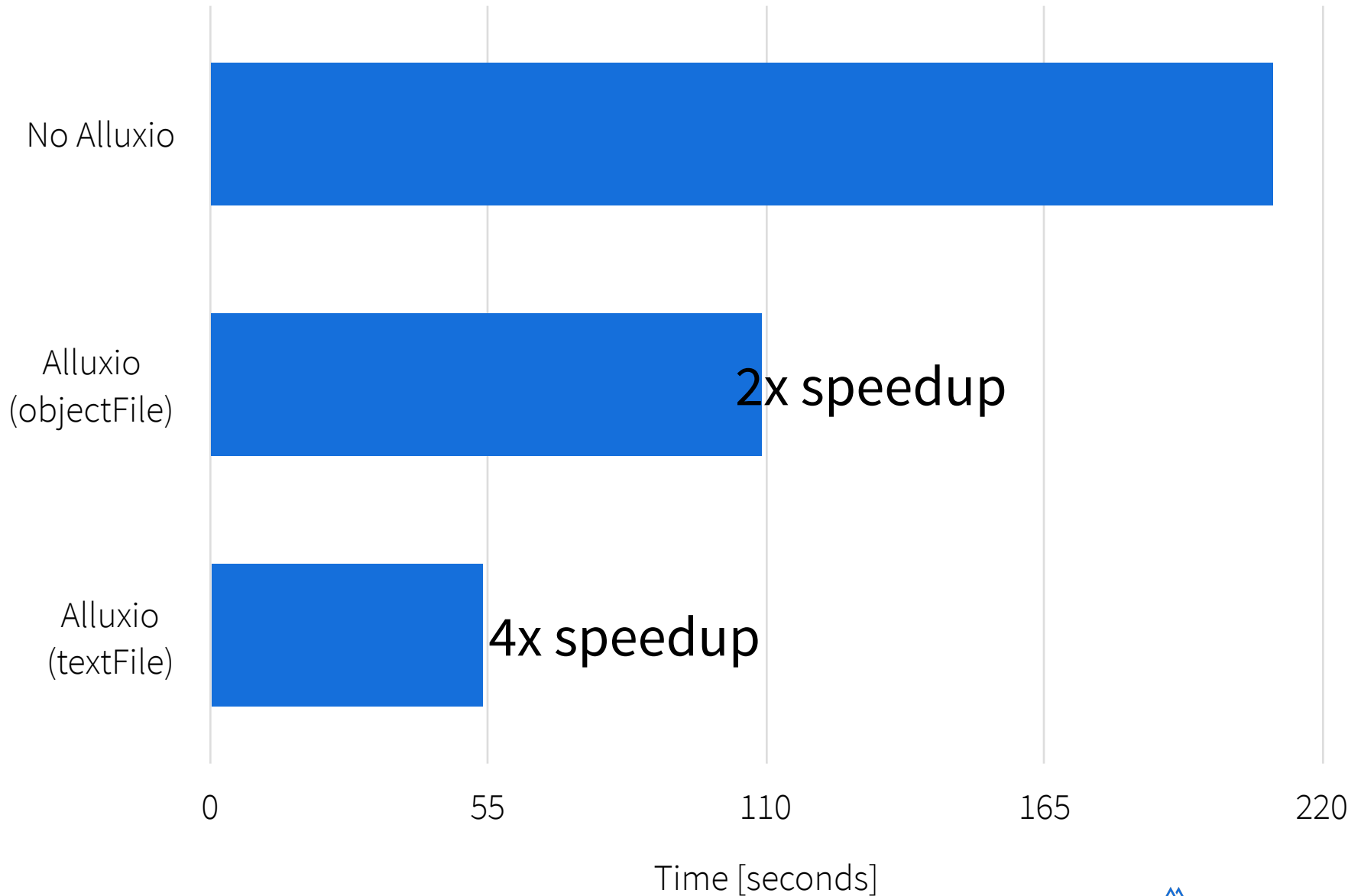
Single r3.2xlarge instance (61GB RAM)

Comparisons:
- Alluxio
- Spark Storage Level: MEMORY_ONLY
- Spark Storage Level: MEMORY_ONLY_SER
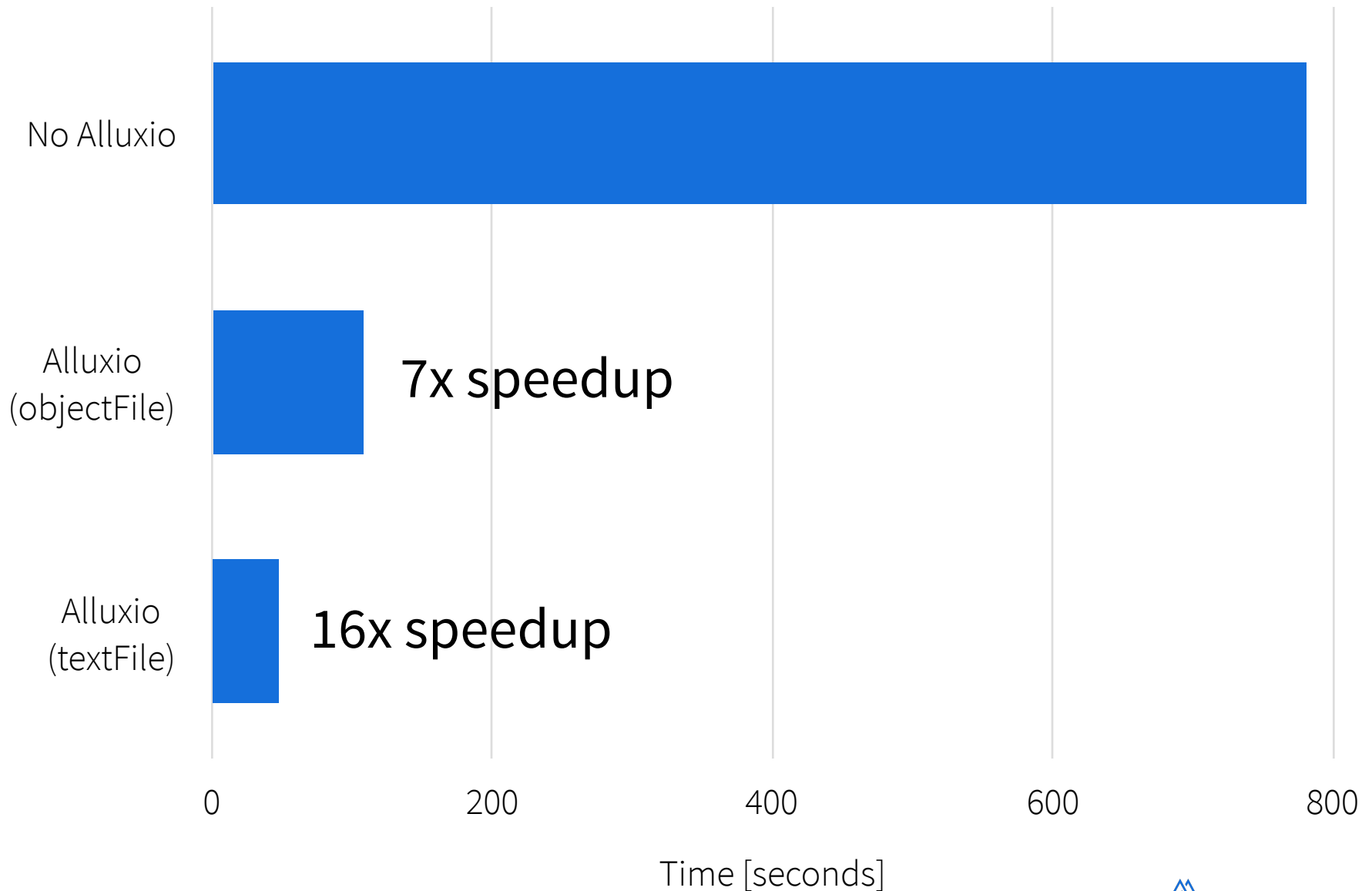- Spark Storage Level: DISK_ONLY

ALLUXIO

Reading Cached RDD

Time [seconds] (y-axis): 0, 75, 150, 225, 300

RDD Size [GB] (x-axis): 0, 13, 25, 38, 50

Legend:
- Alluxio (textFile)
- Alluxio (objectFile)
- DISK_ONLY
- MEMORY_ONLY_SER
- MEMORY_ONLY

ALLUXIO

New Context: Read 50 GB RDD (SSD)

No Alluxio

Alluxio (objectFile) — 2x speedup

Alluxio (textFile) — 4x speedup

Time [seconds]

0   55   110   165   220

ALLUXIO

New Context: Read 50 GB RDD (S3)

	Time [seconds]
No Alluxio	(bar extends to ~780)
Alluxio (objectFile)	7x speedup
Alluxio (textFile)	16x speedup

0   200   400   600   800

Time [seconds]

ALLUXIO

Reading CACHED DATAFRAME (parquet)

Time [seconds] vs DataFrame Size [GB]

Legend:
- (TextFile)
- DISK_ONLY
- MEMORY_ONLY_SER
- MEMORY_ONLY

ALLUXIO

New Context: Read 50 GB DATAFRAME (SSD)

2.5x speedup

Time [seconds]

ALLUXIO

New CONTEXT: Read 50 GB DataFrame (S3)

Bar chart comparing read time for "No Alluxio" (~950 seconds) versus "Alluxio" (~100 seconds), with X-axis "Time [seconds]" ranging from 0 to 1750.

10x speedup (avg), 17x speedup (peak)
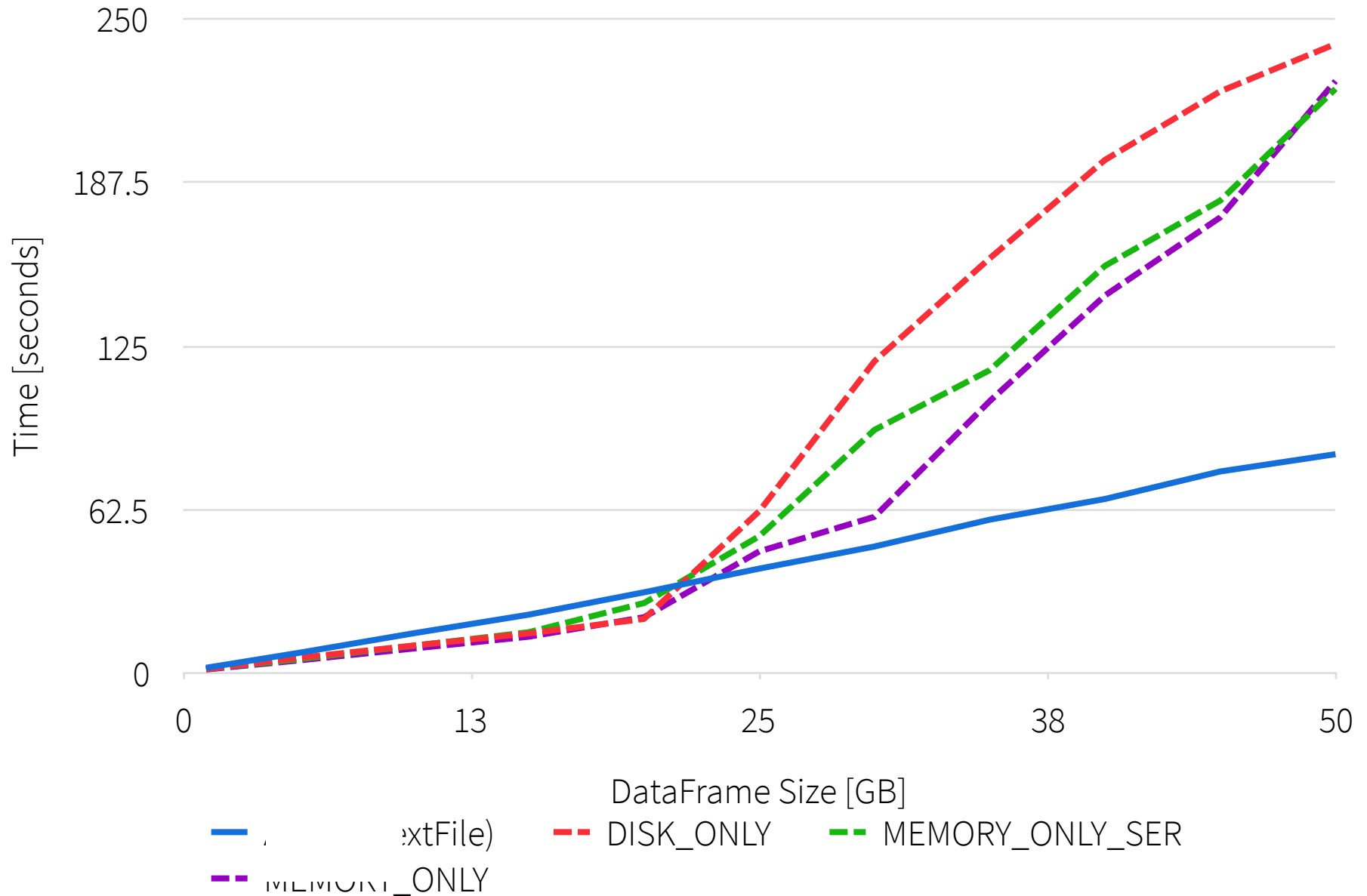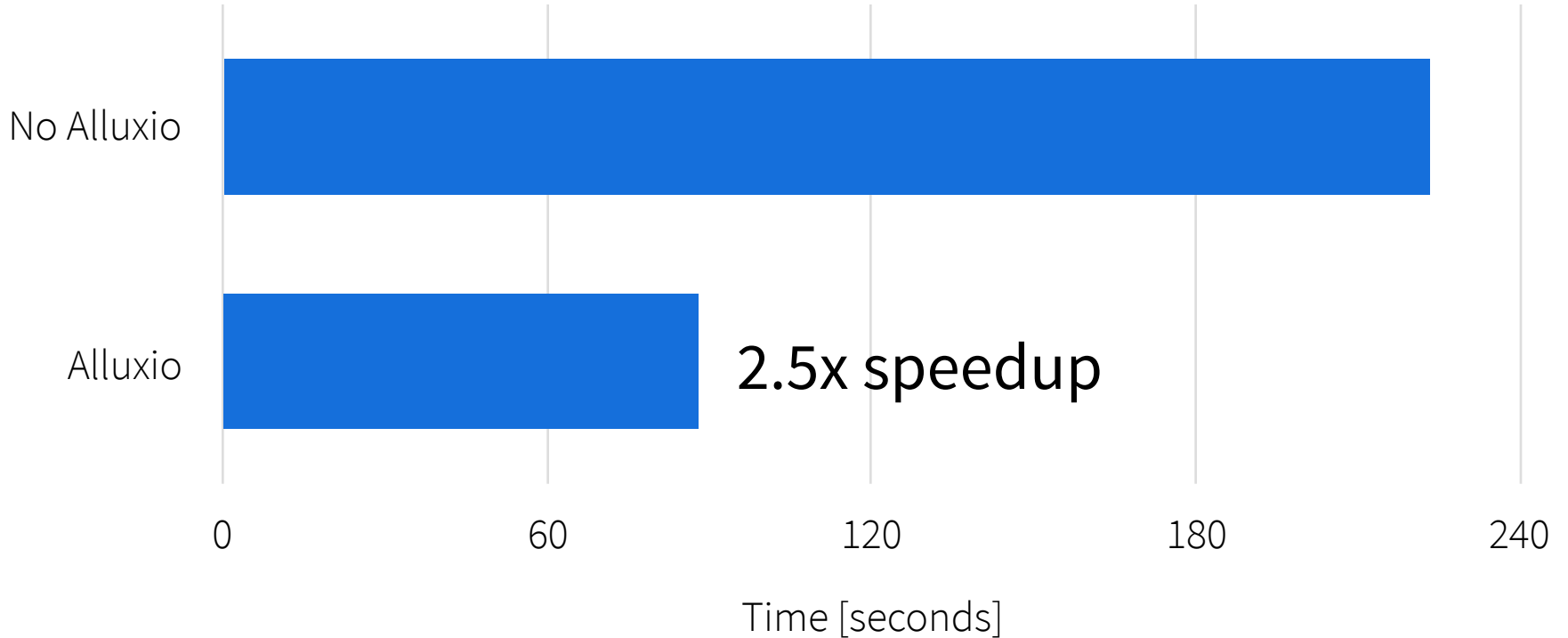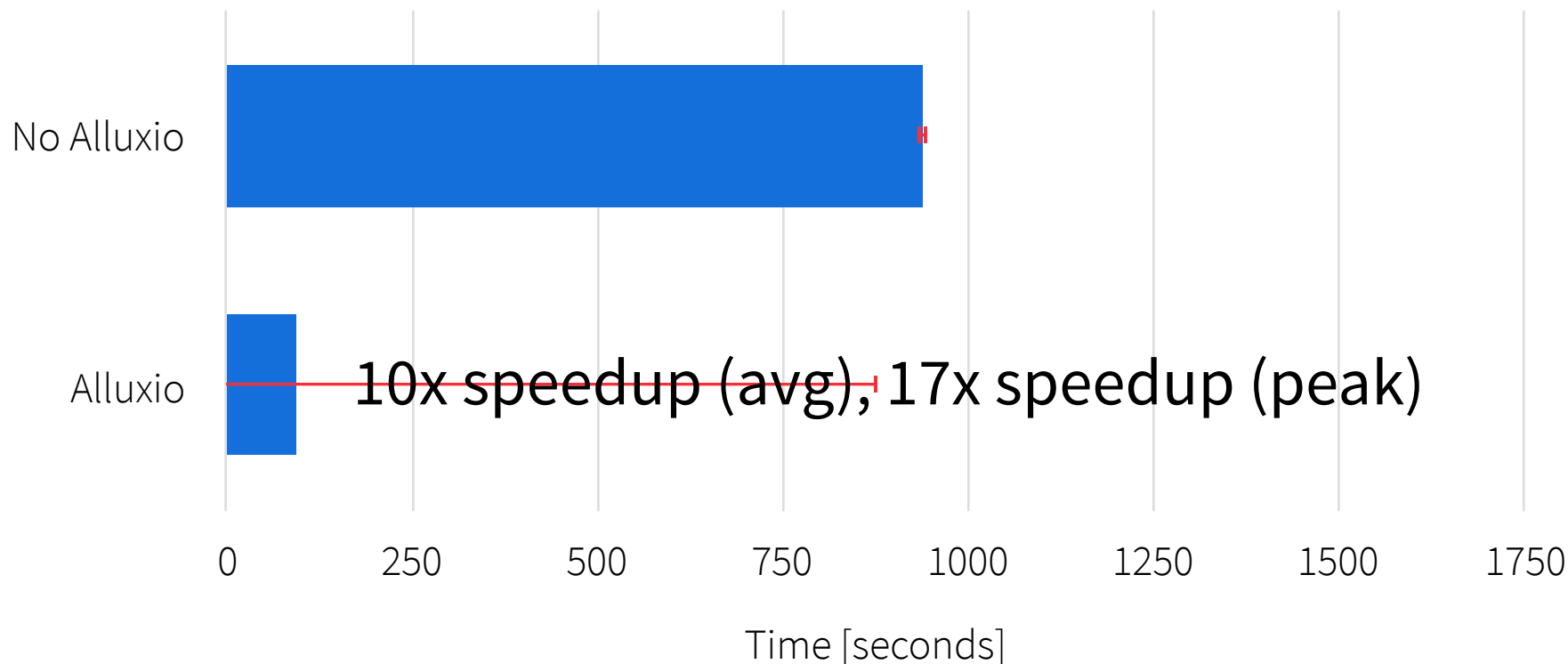
ALLUXIO

CONCLUSION

- Easy to use with Spark

- Better Performance

- Predictable Performance

ALLUXIO

Thank you!

Contact: jiri@alluxio.com

Twitter: @jsimsa

Websites: www.alluxio.com and www.alluxio.org

ALLUXIO