A Reliable Memory-Centric Distributed Storage System

Haoyuan Li
October 16 @ Strata & Hadoop World NYC

Website: tachyon-project.org
Meetup: www.meetup.com/Tachyon
Outline

• Overview
  – Feature 1: Memory Centric Storage Architecture
  – Feature 2: Lineage in Storage

• Open Source

• Roadmap
Outline

• **Overview**
  – Feature 1: Memory Centric Storage Architecture
  – Feature 2: Lineage in Storage

• **Open Source**

• **Roadmap**
Projects

- Design next generation data analytics stack: Berkeley Data Analytics Stack (BDAS)
  - a cluster manager making it easy to write and deploy distributed applications.

- Spark
  - a parallel computing system supporting general and efficient in-memory execution.

- TACHYON
  - a reliable distributed memory-centric storage enabling memory-speed data sharing.
Why Tachyon?
Memory is **King**

- RAM throughput increasing **exponentially**
- Disk throughput increasing **slowly**

*Memory-locality key to interactive response time*
Realized by many...

- Frameworks already leverage memory

April 7, 2012

Many kinds of memory-centric data management

I'm frequently asked to generalize in some way about in-memory or memory-centric data management. I can start:

- The desire for human real-time interactive response naturally leads to...
Problem solved?
Missing a Solution for Storage Layer
An Example: Spark

- Fast in-memory data processing framework
  - Keep one in-memory copy inside JVM
  - Track lineage of operations used to derive data
  - Upon failure, use lineage to recompute data
Data Sharing is the bottleneck in analytics pipeline: Slow writes to disk

storage engine & execution engine same process (slow writes)
Issue 1

Data Sharing is the bottleneck in analytics pipeline:
Slow writes to disk

storage engine & execution engine same process (slow writes)
Issue 2

Cache loss when process crashes.

execution engine & storage engine same process
Issue 2

Cache loss when process crashes.

execution engine & storage engine same process
Issue 2

Cache loss when process crashes.

execution engine & storage engine same process

HDFS / Amazon S3
Issue 3

In-memory Data Duplication & Java Garbage Collection

execution engine & storage engine same process (duplication & GC)
Tachyon

*Reliable* data sharing at *memory-speed* within and across cluster frameworks/jobs
Solution Overview

Basic idea

• Feature 1: memory-centric storage architecture
• Feature 2: push lineage down to storage layer

Facts

• One data copy in memory
• Recomputation for fault-tolerance
Stack

Computation Frameworks
(Spark, MapReduce, Impala, H2O, ...)

Tachyon

Existing Storage Systems
(HDFS, S3, GlusterFS, ...)

Memory-Centric Storage Architecture
Issue 1 revisited

Memory-speed data sharing among jobs in different frameworks

execution engine & storage engine same process (fast writes)
Issue 2 revisited

Keep in-memory data safe, even when a job crashes.

execution engine & storage engine same process

Spark Task

Spark memory block manager

Tachyon in-memory
Issue 2 revisited

*Keep in-memory data safe, even when a job crashes.*

execution engine &
storage engine
same process

- Spark memory
  - block manager

- Tachyon
  - in-memory

- HDFS / Amazon S3
Issue 2 revisited

Keep in-memory data safe, even when a job crashes.

execution engine & storage engine same process

Tachyon in-memory

HDFS / Amazon S3

crash
Issue 3 revisited

No in-memory data duplication, much less GC

execution engine & storage engine same process (no duplication & GC)
Lineage in Storage (alpha)
Comparison with in Memory HDFS

Write Throughput

- Tachyon Write
- MemHDFS Write
- Theoretical Replication (2 copies) Based Write

Throughput (GB/Sec)

Number of Machines

0 10 20 30
Performance comparison for realistic workflow. The workflow ran 4x faster on Tachyon than on MemHDFS. In case of node failure, applications in Tachyon still finishes 3.8x faster.
Further Improve Spark’s Performance

Grep Program
How easy / hard to use Tachyon?
Spark/MapReduce/Shark without Tachyon

• Spark
  – val file = sc.textFile("hdfs://ip:port/path")

• Hadoop MapReduce
  – hadoop jar hadoop-examples-1.0.4.jar wordcount hdfs://localhost:19998/input hdfs://localhost:19998/output

• Shark
  – CREATE TABLE orders_cached AS SELECT * FROM orders;
Spark/MapReduce/Shark with Tachyon

• Spark
  – val file = sc.textFile("tachyon://ip:port/path")

• Hadoop MapReduce
  – hadoop jar hadoop-examples-1.0.4.jar wordcount tachyon://localhost:19998/input tachyon://localhost:19998/output

• Shark
  – CREATE TABLE orders_tachyon AS SELECT * FROM orders;
Spark on Tachyon

./bin/spark-shell
sc.hadoopConfiguration.set("fs.tachyon.impl", "tachyon.hadoop.TFS")

// Load input from Tachyon
val file = sc.textFile("tachyon://localhost:19998/LICENSE")
file.count() ; file.take(10);

// Store RDD OFF_HEAP in Tachyon
import org.apache.spark.storage.StorageLevel;
file.persist(StorageLevel.OFF_HEAP)
file.count(); file.take(10);

// Save output to Tachyon
file.flatMap(line => line.split(" ")).map(s => (s, 1)).reduceByKey((a, b) => a + b).saveAsTextFile("tachyon://localhost:19998/LICENSE_WC")
Outline

• Overview
  – Feature 1: Memory Centric Storage Architecture
  – Feature 2: Lineage in Storage

• Open Source

• Roadmap
History

Started at UC Berkeley AMPLab from the summer of 2012

- Reliable, Memory Speed Storage for Cluster Computing Frameworks (UC Berkeley EECS Tech Report)
- Haoyuan Li, Ali Ghodsi, Matei Zaharia, Ion Stoica, Scott Shenker
• Apache License 2.0, Version 0.5.0 (July 2014)

• Deployed at tens of companies

• 20+ Companies Contributing

• Spark and MapReduce applications can run without any code change
Release Growth

Tachyon 0.1:
-1 contributor

Dec '12
Release Growth

- Tachyon 0.1: -1 contributor (Dec '12)
- Tachyon 0.2: -3 contributors (Apr '13)
Release Growth

Tachyon 0.1:
- 1 contributor
Dec ’12

Tachyon 0.2:
- 3 contributors
Apr ’13

Tachyon 0.3:
- 15 contributors
Oct’13
Release Growth

Tachyon 0.1: -1 contributor
Dec '12

Tachyon 0.2: -3 contributors
Apr '13

Tachyon 0.3: -15 contributors
Oct '13

Tachyon 0.4: -30 contributors
Feb '14

- 3 contributors
Feb '14

- 15 contributors
Oct '13

- 3 contributors
Apr '13

-1 contributor
Dec '12
Release Growth

Tachyon 0.1:
- 1 contributor
Dec '12

Tachyon 0.2:
- 3 contributors
Apr '13

Tachyon 0.3:
- 15 contributors
Oct '13

Tachyon 0.4:
- 30 contributors
Feb '14

Tachyon 0.5:
- 46 contributors
July '14
Open Community

- Berkeley Contributors
- Non-Berkeley Contributors
Thanks to our Code Contributors!

Aaron Davidson  David Zhu  Lukasz Jastrzebski  Sean Zhong
Achal Soni  Du Li  Manu Goyal  Seonghwan Moon
Ali Ghodsi  Fei Wang  Mark Hamstra  Shivaram Venkataraman
Andrew Ash  Gerald Zhang  Mingfei Shi  Srinivas Parayya
Anurag Khandelwal  Grace Huang  Mubarak Seyed  Tao Wang
Aslan Bekirov  Haoyuan Li  Nick Lanham  Timothy St. Clair
Bill Zhao  Henry Saputra  Orcun Simsek  Thu Kyaw
Brad Childs  Hobin Yoon  Pengfei Xuan  Vamsi Chitters
Calvin Jia  Huamin Chen  Qianhao Dong  Xi Liu
Chao Chen  Jey Kottalam  Qifan Pu  Xiang Zhong
Cheng Chang  Joseph Tang  Raymond Liu  Xiaomin Zhang
Cheng Hao  Juan Zhou  Reynold Xin  Zhao Zhang
Colin Patrick McCabe  Jun Aoki  Robert Metzger
David Capwell  Lin Xing  Rong Gu

Logos of various companies and organizations.

43
Thanks to Redhat!

Tachyon is in Fedora 20

Thanks to Redhat!
Commerially supported by Atigeo™ and running in dozens of their customers’ clusters.
Thanks to Redhat!

Tachyon is the Default Off-Heap Storage Solution for Spark
Exchange Data Between Spark and H20

Today, data gets parsed and exchanged between Spark and H2O via Tachyon. Users can interactively query big data both via SQL and ML from within the same context.
Believe from Industry

The Future Architecture of a Data Lake: In-memory Data Exchange Platform Using Tachyon and Apache Spark

Pivotal and EMC are betting on Spark cousin Tachyon as in-memory file system

Pivotal bets on Tachyon as next in-memory file system

Pivotal Expands on Data Lake Vision with Embrace of Project Tachyon
Reaching wider communities: e.g. GlusterFS

**Glusterfs and Tachyon**

*Tachyon*, an in-memory distributed filesystem, is among the most dynamic projects in big data analytics stack. It provides Java io like API, support *Apache Spark*, and vastly improves Spark’s performance under large data set.

![Diagram showing the relationship between Tachyon, Spark, Hadoop MR, H2O, and other systems such as HDFS, S3, Glusterfs, and Posix.](blog.gluster.org/2014/08/glusterfs-and-tachyon/)

*by huamin on August 7, 2014*
Under Filesystem Choices (Big Data, Cloud, HPC, Enterprise)
Under Filesystem Choices (Big Data, Cloud, HPC, Enterprise)
Outline

• Overview
  – Feature 1: Memory Centric Storage Architecture
  – Feature 2: Lineage in Storage

• Open Source

• Roadmap
Features

• Memory Centric Storage Architecture
• Lineage in Storage (alpha)
• Hierarchical Local Storage
• Data Serving
• Different hardware
• More...
• Your Requirements?
Short Term Roadmap (0.6 Release)

• Ceph Integration (Ceph Community, Redhat)

• Hierarchical Local Storage (Intel)

• Performance Improvement (Yahoo)

• Multi-tenancy (AMPLab)

• Mesos Integration (Mesos Community, Mesosphere)

• Network Sub-system Improvement (Pivotal)

• Many more from AMPLab and Industry Contributors
Goal?
Better Assist Other Components

Welcome Collaboration!
Thanks!
Questions?

• More Information:
  – Website: http://tachyon-project.org
  – Github: https://github.com/amplab/tachyon
  – Meetup: http://www.meetup.com/Tachyon
• Email: haoyuan@cs.berkeley.edu
Release Growth

- Tachyon 0.1: -1 contributor
- Tachyon 0.2: -3 contributors
- Tachyon 0.3: -15 contributors
- Tachyon 0.4: -30 contributors
- Tachyon 0.5: -46 contributors

Dec '12
Apr '13
Oct '13
Feb '14
July '14