QCon 全球软件开发大会 INTERNATIONAL SOFTWARE DEVELOPMENT CONFERENCE

BEIJING 2018

使用开源分布式存储系统Alluxio来有效的分离计算与存储

演讲者: 范斌

binfan@alluxio.com, Alluxio创始成员, 资深架构师

微信公众号

Geekbang. InfoQ

GTLC
GLOBAL
TECH LEADERSHIP
CONFERENCE
全球技术领导力峰会

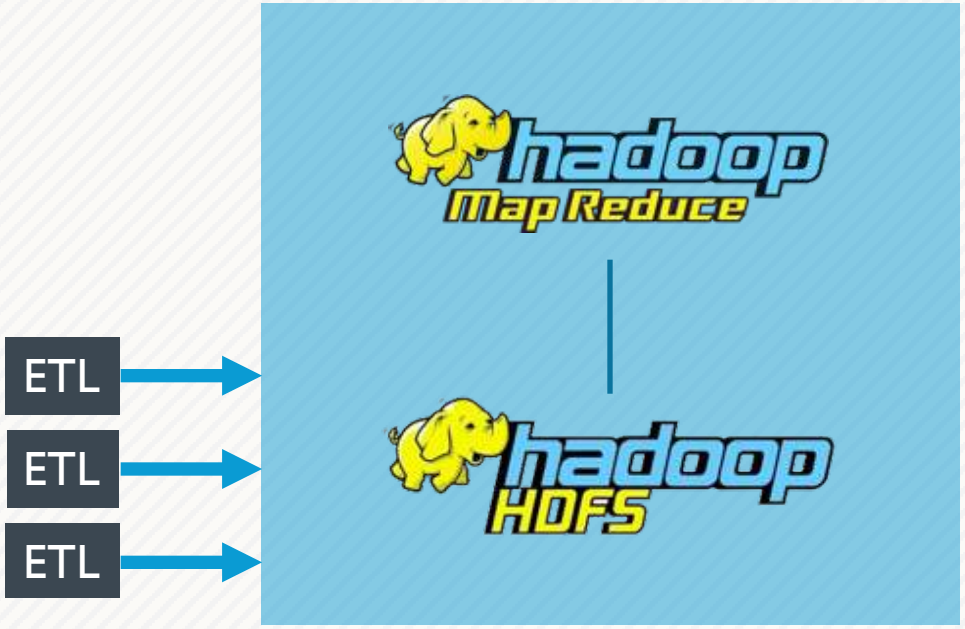
通往**年薪百万**的CTO的路上，
如何打造自己的技术**领导**力？

扫描二维码了解详情

Outline

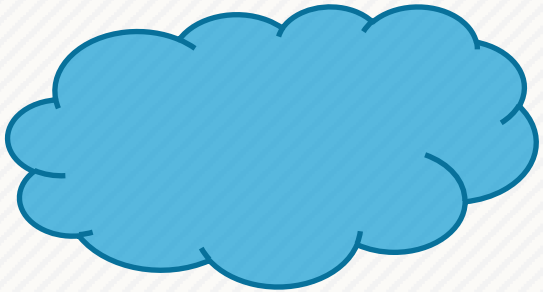
1. Why we built Alluxio
2. Alluxio's innovations
3. Alluxio's Architecture
4. What's new in 1.7

Data Ecosystem A Decade Ago



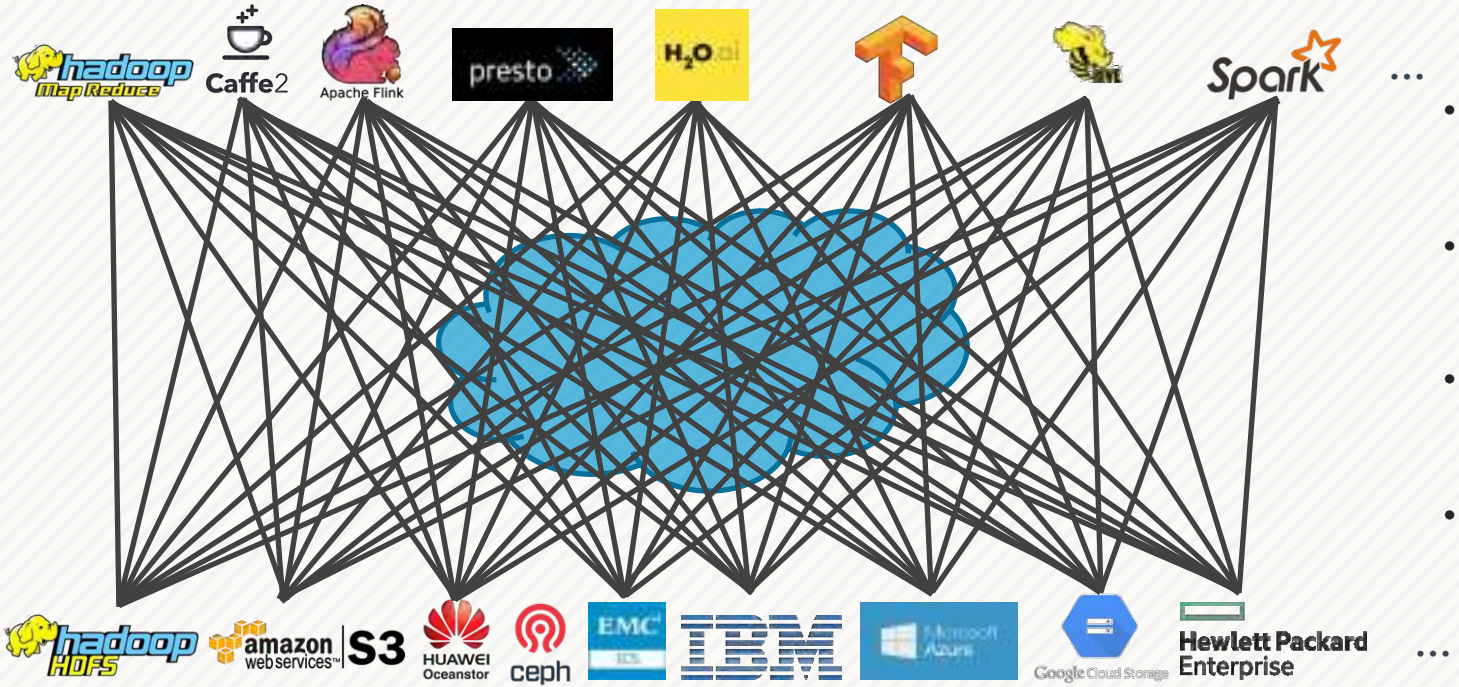
- One Compute Framework
- Single Storage System
- Co-located

Data Ecosystem Today



- Many Compute Frameworks
- Many Storage Systems
- Most not co-located

Data Ecosystem Issues



- Each app manages multiple data sources
- Data source changes require global updates
- Storage optimizations requires app change
- Poor performance due to lack of locality

Data Ecosystem Challenges

Heavy integrations create painful organizational drag

1 Speed & Complexity

- Integration and interoperability issues (on prem, hybrid, cloud)
- Many departments & groups

2 Data Freshness

- Cross-network movement is slow
- Copies create lag
- Data quality suffers with copies

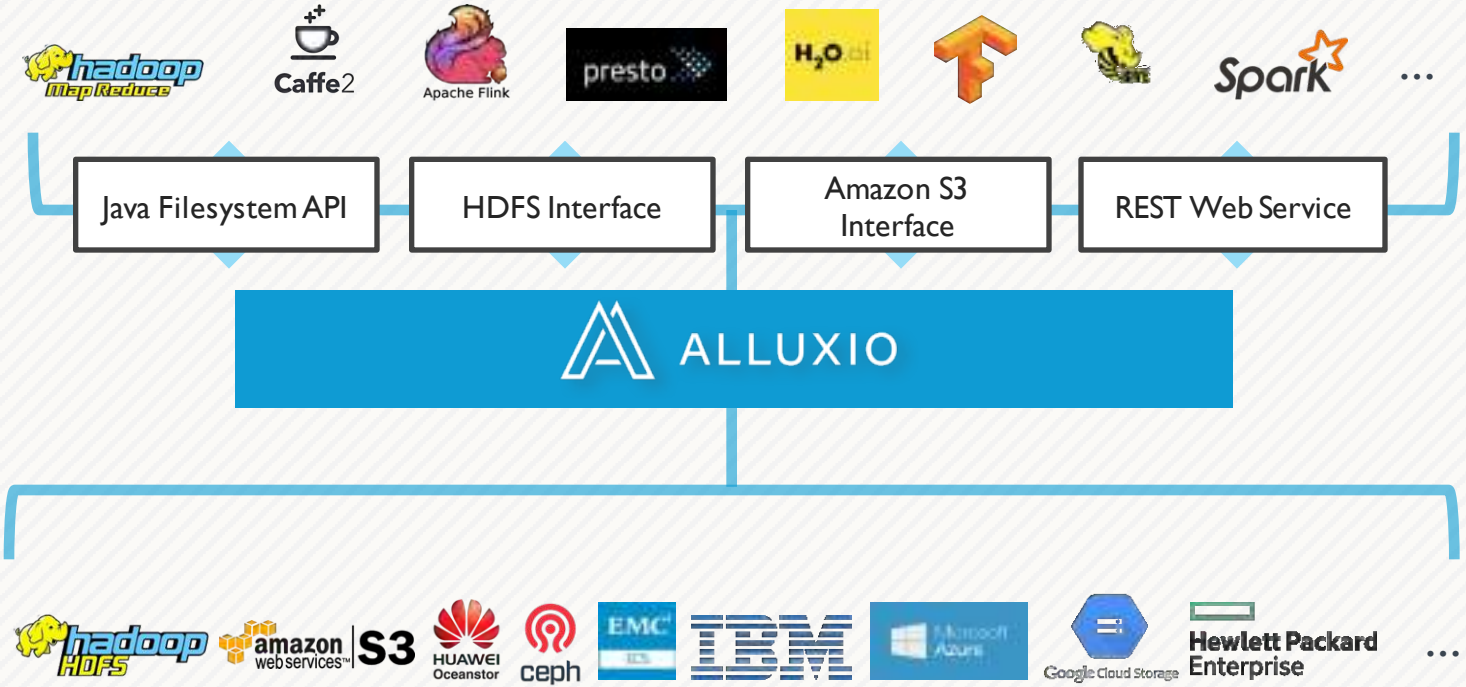
3 Cost

- Many-to-many integrations are expensive
- Data duplication

4 Security & Governance

- Data security & governance is increasingly complex

Data Ecosystem with Alluxio



- Apps only talk to Alluxio
- Simple Add/Remove
- No App Changes
- Highest performance in Memory

Alluxio Design Principles

1 Big Data & Machine Learning

- Interoperability with leading projects
- Large scale data sets
- High IO

2 Data Sharing

- Don't own the data
- Multiple apps sharing common data
- Data stored in multiple, hybrid systems

3 High Speed Data Access

- Remote data
- Hot/warm/cold data
- Temporary data
- Read/write support

4 Enterprise Class

- Distributed architecture
- Commodity hardware
- Service-oriented
- High availability
- Security

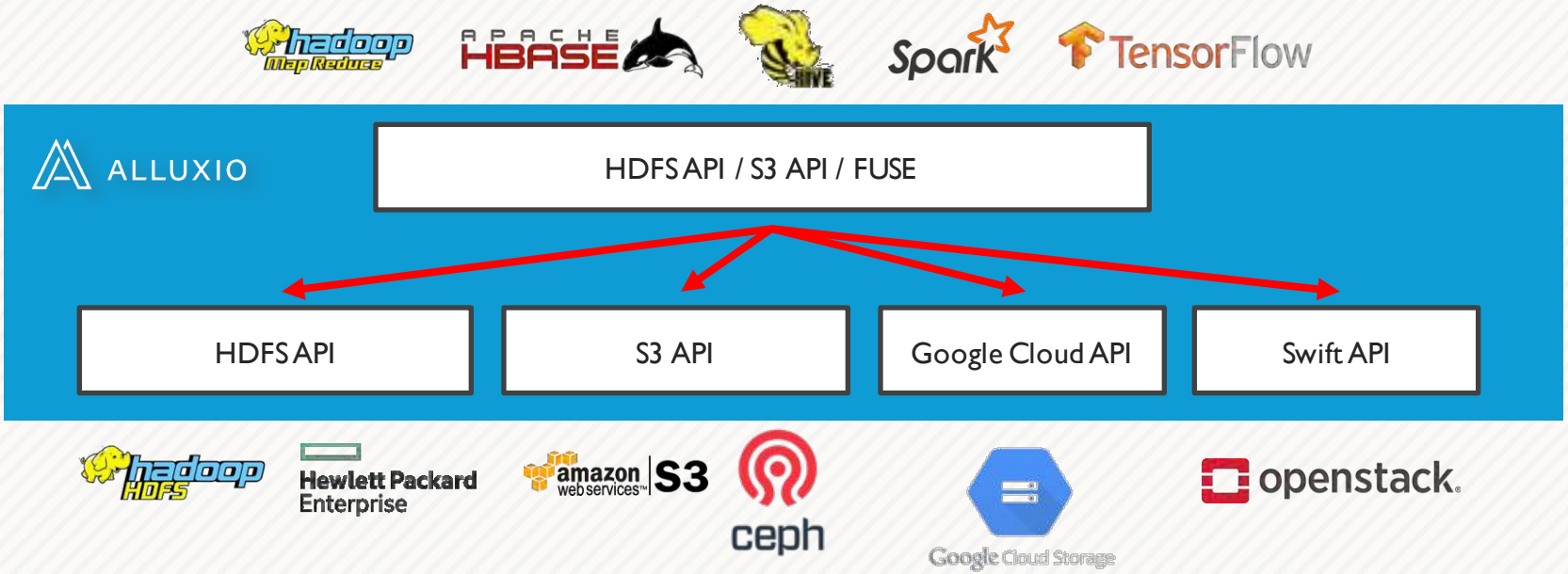
Outline

1. Why we built Alluxio
2. Alluxio's innovations
3. Alluxio's Architecture
4. What's new in 1.7

Alluxio Innovation:

Server-side API Translation

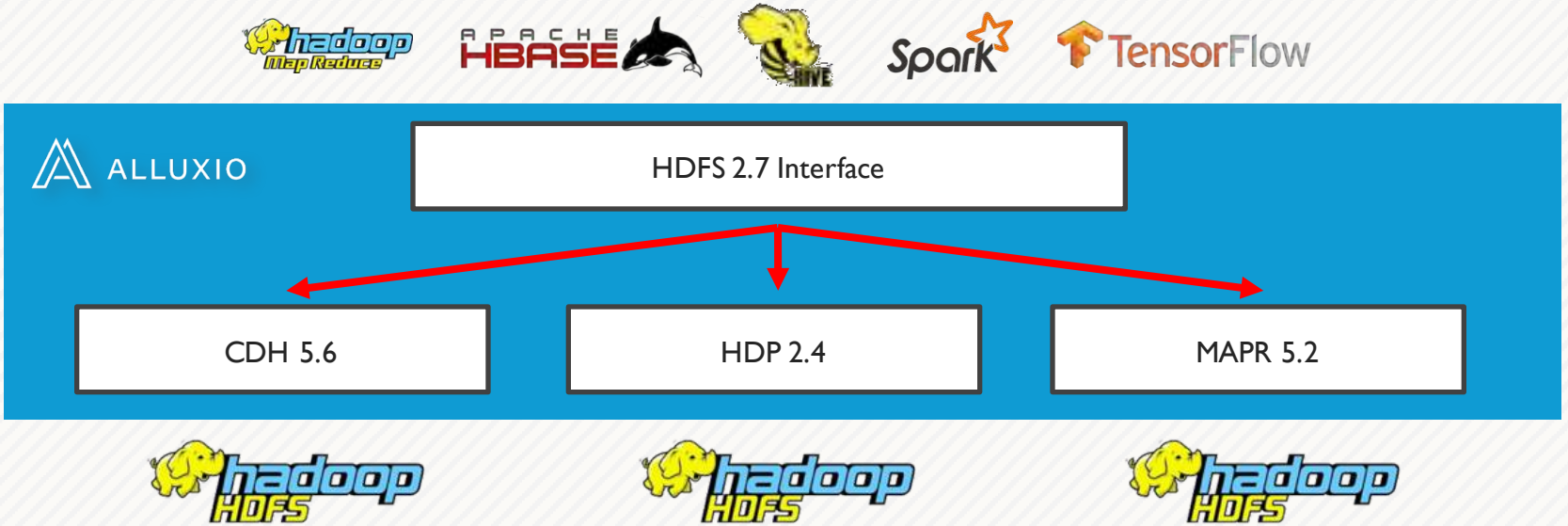
Convert from Client-side Interface to Native Storage Interface



Alluxio Innovation:

Server-side API Translation

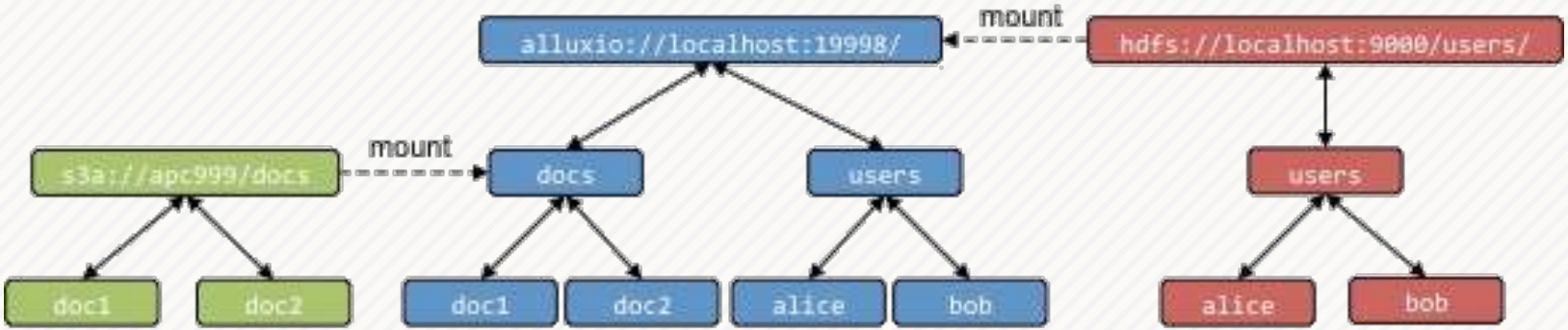
Convert between different versions of HDFS



Unified Namespace

Enables effective data management across different Under Stores

- Uses Mounting with Transparent Naming



Alluxio Innovation:

Unified Namespace

Create a catalog of available data sources for Data Scientists

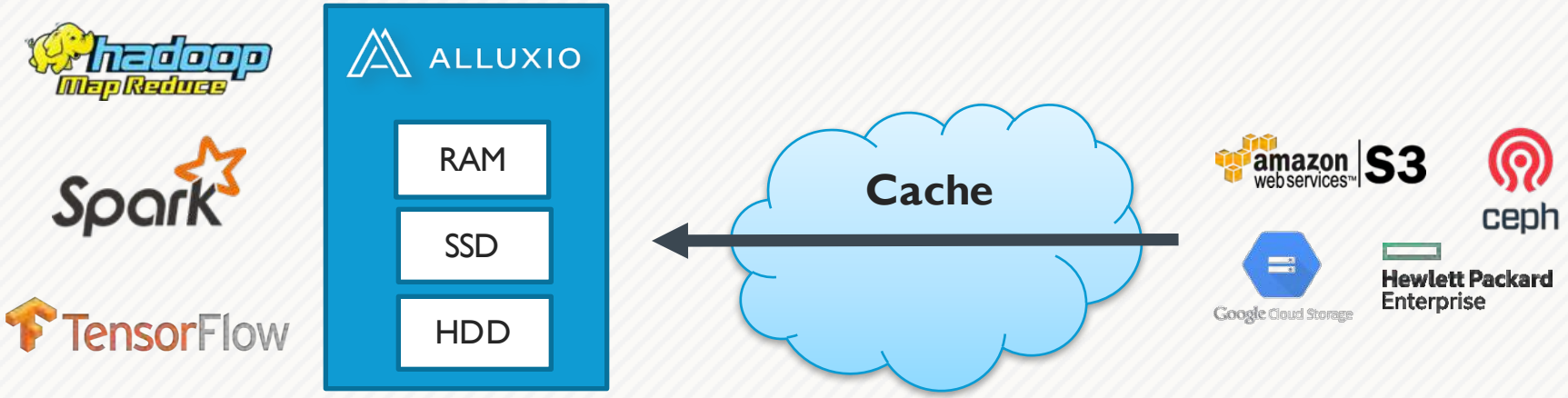
alluxio://

```
/finance/customer-transactions/
/finance/vendor-transactions/
/operations/device-logs/
/operations/phone-call-recordings/
/operations/check-images/
/research/us-economic-data/
/research/intl-economic-data/
/marketing/advertising-dataset/
/marketing/marketing-funnel-dataset/
```

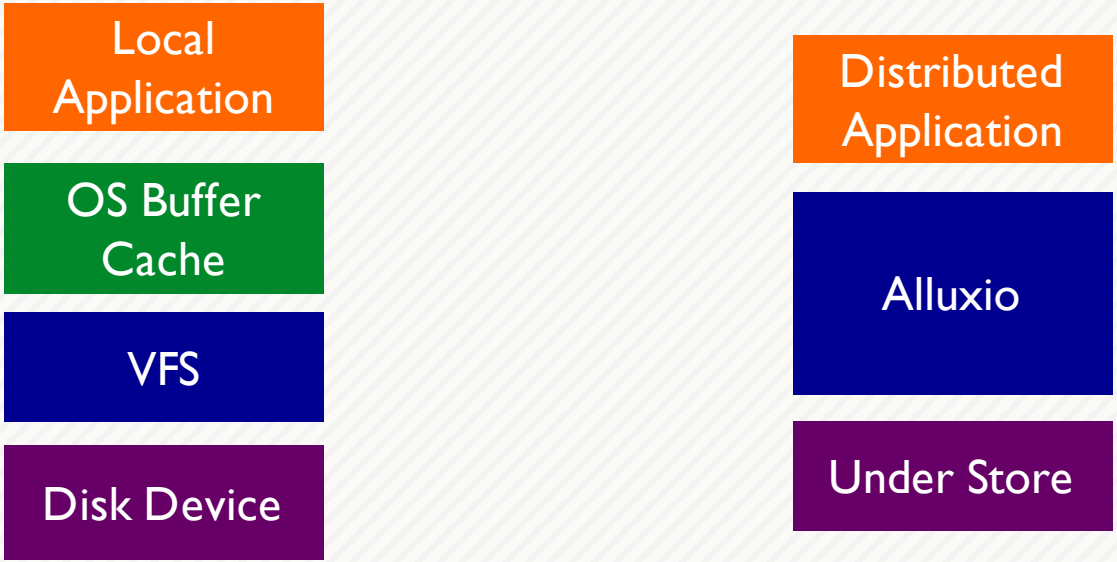
Alluxio Innovation:

Intelligent Cache

Local performance from remote data using native multi-tier storage



Analogy to A Single Machine Stack



Where to use Alluxio

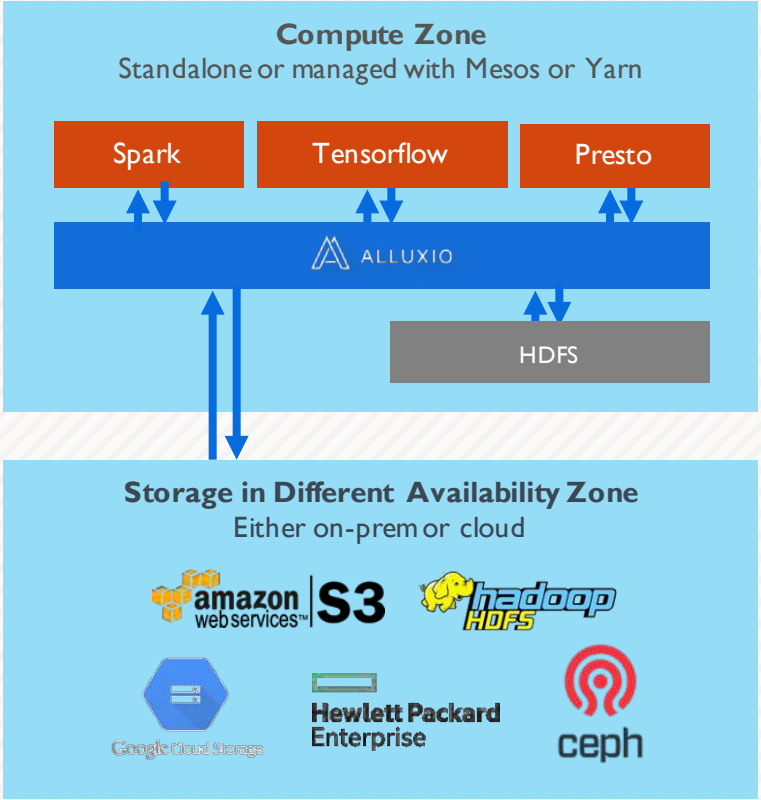
Finding high-fit Alluxio use-cases

Guidelines

- ✓ Cloud deployment
- ✓ Compute separated from storage
- ✓ I/O or network latency exists
- ✓ Unification of many storage systems
- ✓ Applications sharing long lived data

More checks result in higher fit applications

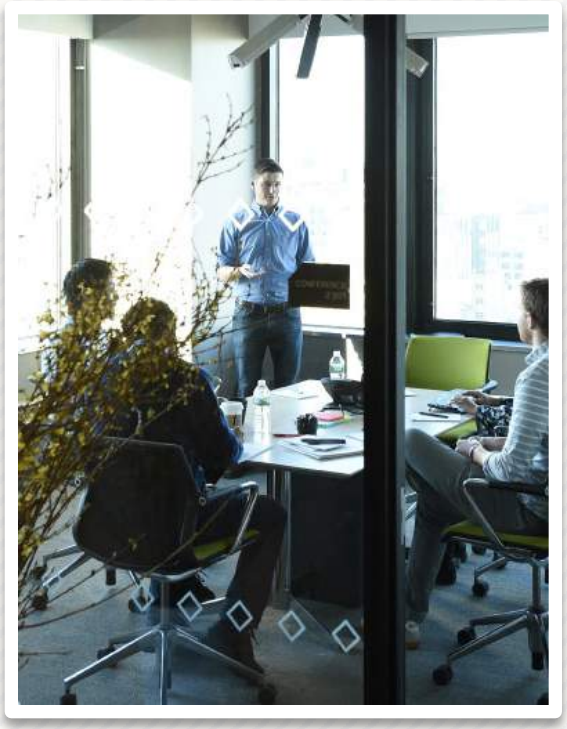
Alluxio is installed with or near compute to unify data stores, stage remote data, and improve system performance.

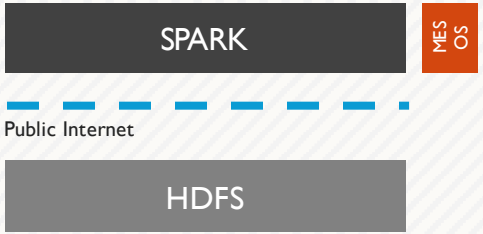


100+ known production deployments

AND MORE!

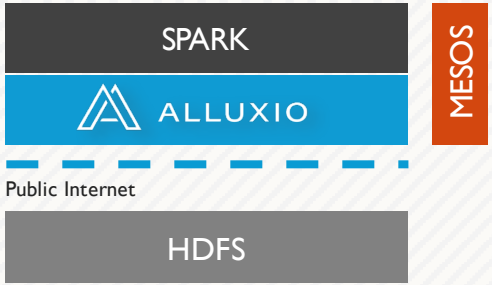
Machine Learning Case Study – TWO SIGMA





Challenge –
Slow training of model for algorithmic trading in $46B data driven Hedge Fund

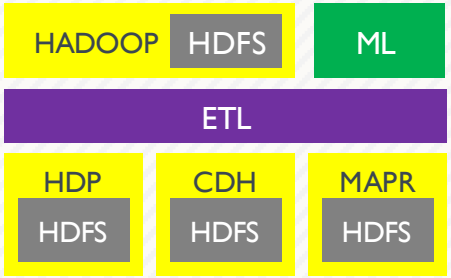
Data access was slow, costing them $$ in compute cost and lower modeler productivity



Solution –
With Alluxio, data access are 10-30X faster

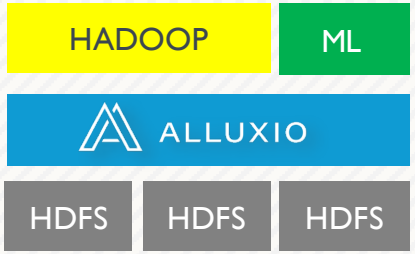
Impact –
Increased efficiency on training of ML algorithm, lowered compute cost and increased modeler productivity, resulting in 14 day ROI of Alluxio

Consumer Intelligence Use Case – Top 3 Telco



Challenge –
Desired a central view of consumer information in near real time for proactive support.

Many HDFS, different distributions, many incompatible versions. On-prem & cloud. Integration through heavy ETL.



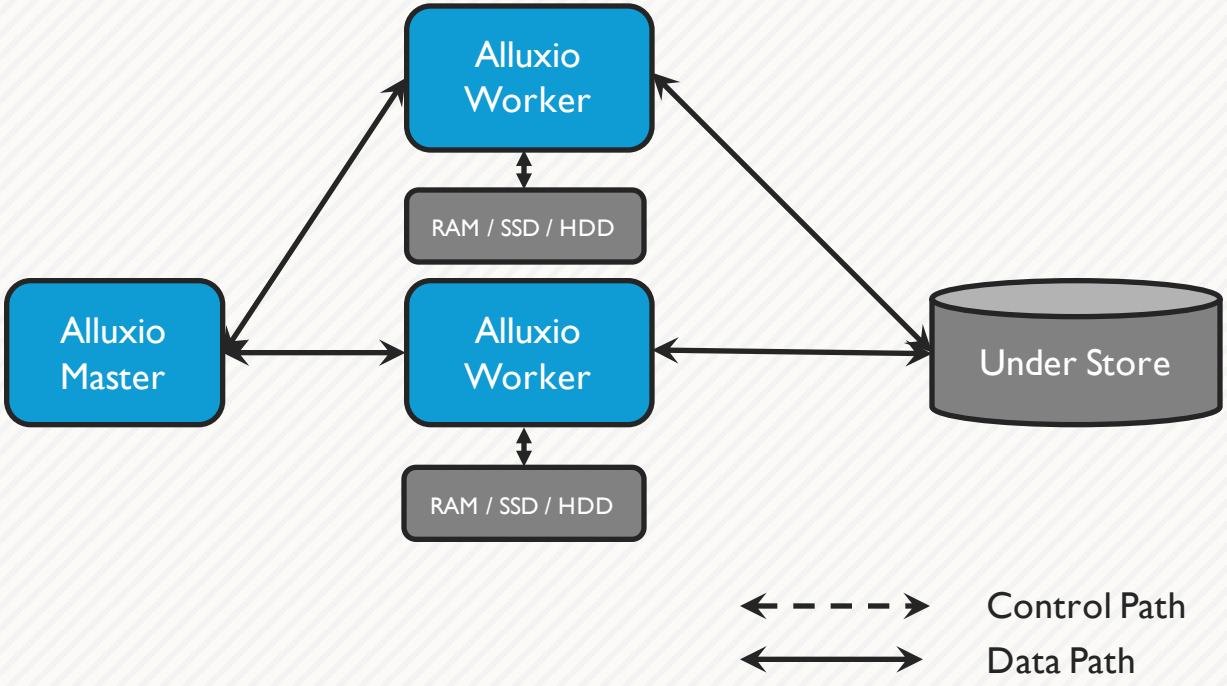
Solution –
Alluxio integrates data into central catalog for fast access to consumer interaction records.

Impact –
Reduced integration time
Faster data speed & freshness

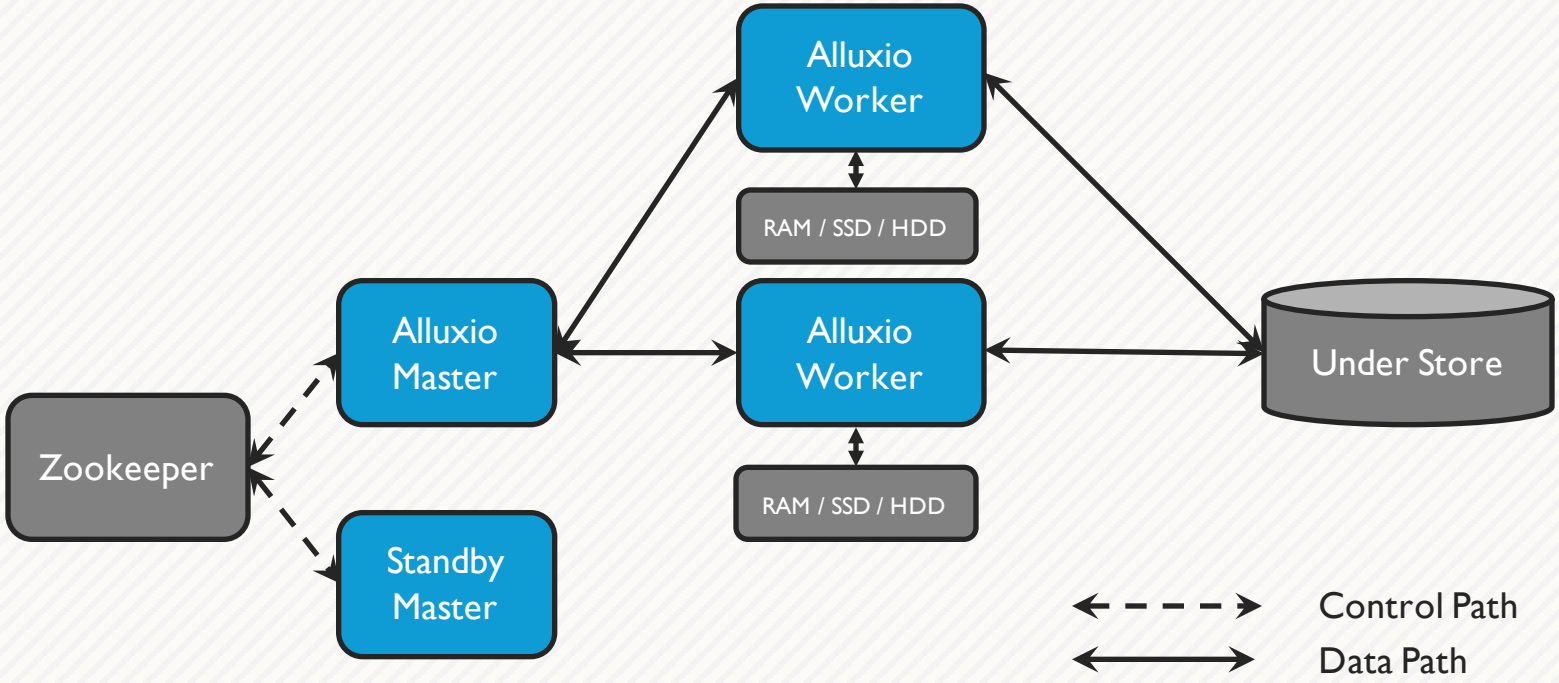
Outline

1. Why we built Alluxio
2. Alluxio's innovations
3. Alluxio's Architecture
4. What's new in 1.7

Alluxio Architecture

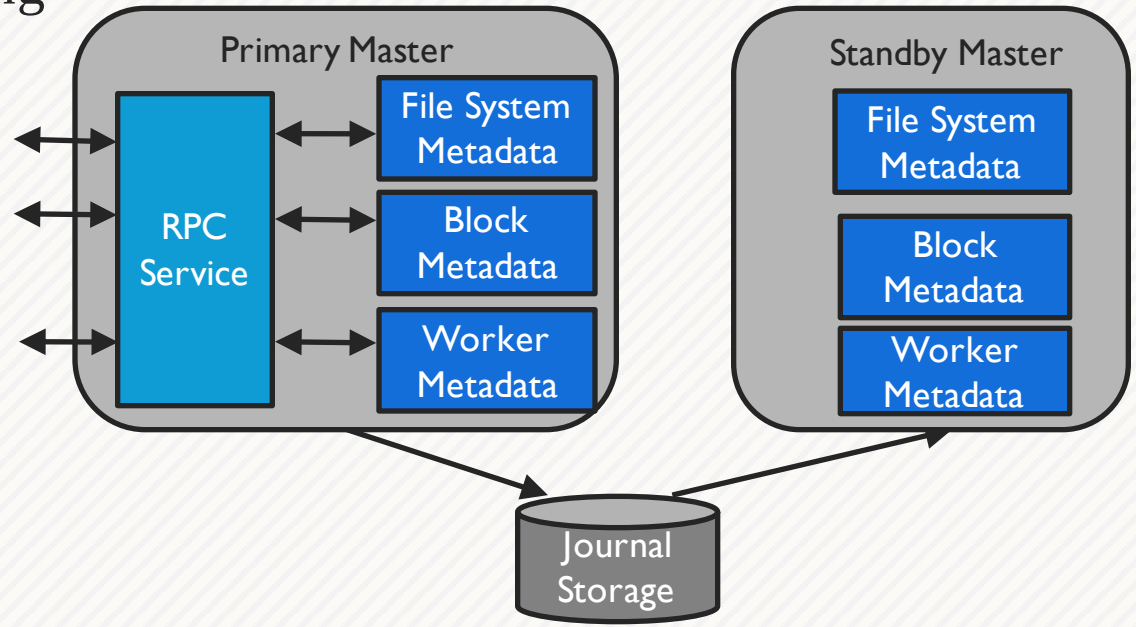


Alluxio Architecture: HA



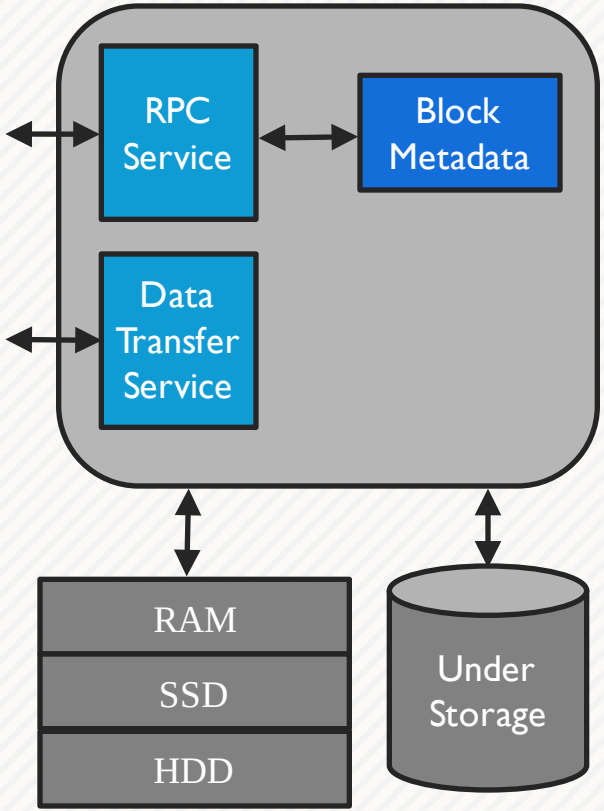
Alluxio Master

- Master responsible for managing metadata
- Standby masters used for journal checkpoints and fault tolerance
- Performs distributed storage metadata operations



Alluxio Worker

- Worker responsible for managing block data
- Each worker manages metadata for the block data it stores
- Workers store block data on various local storage mediums
- Performs distributed storage data operations

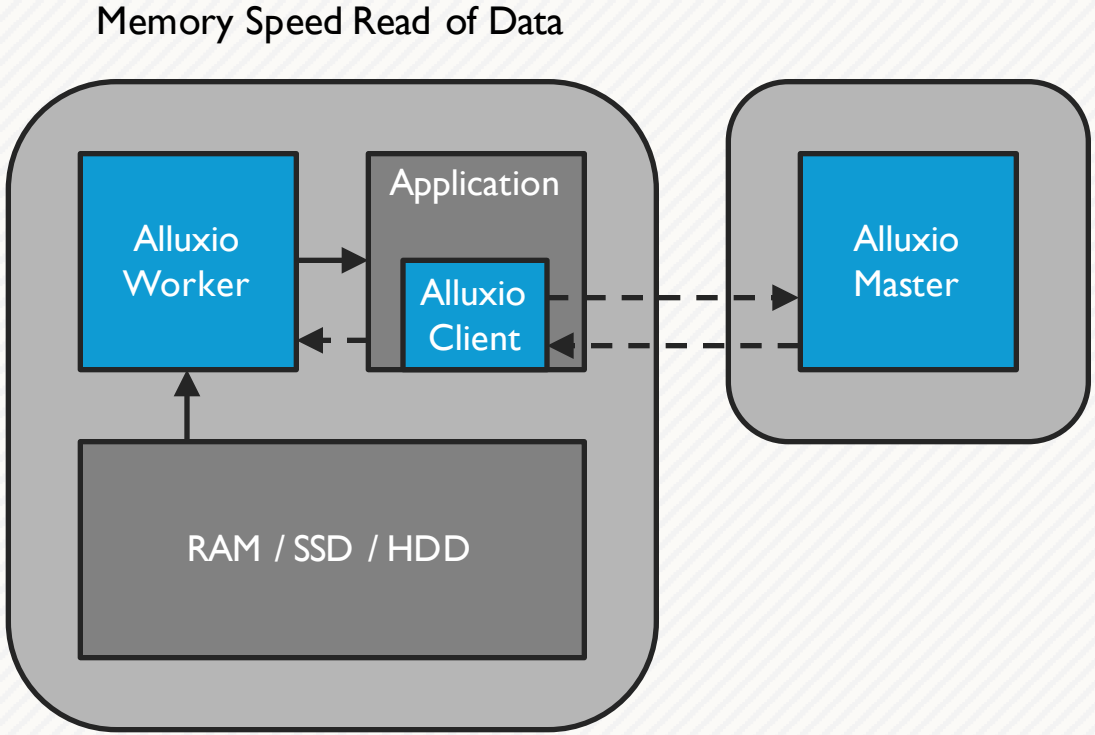


Data Flow In Alluxio

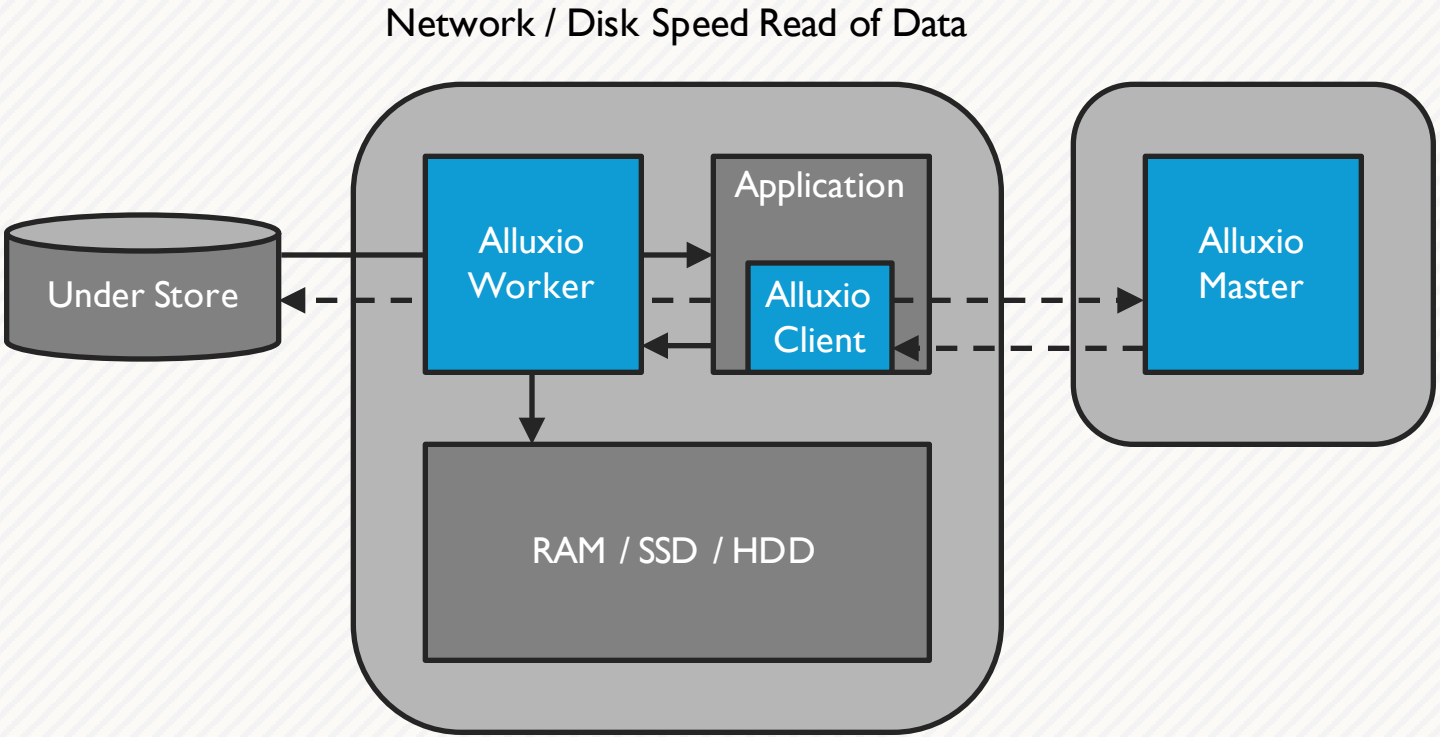
ALLUXIO

- Applications Read/Write data via the Alluxio Client
  - Ideally, Alluxio deployed on same nodes as compute so Alluxio Client and Alluxio Workers on same node
- Different Read Scenarios
  - Read data in Alluxio, on same node as client
  - Read data in Alluxio, not on same node as client
  - Read data not in Alluxio
- Different Write Scenarios
  - Write data only to Alluxio
  - Write data to Alluxio and Under Store synchronously

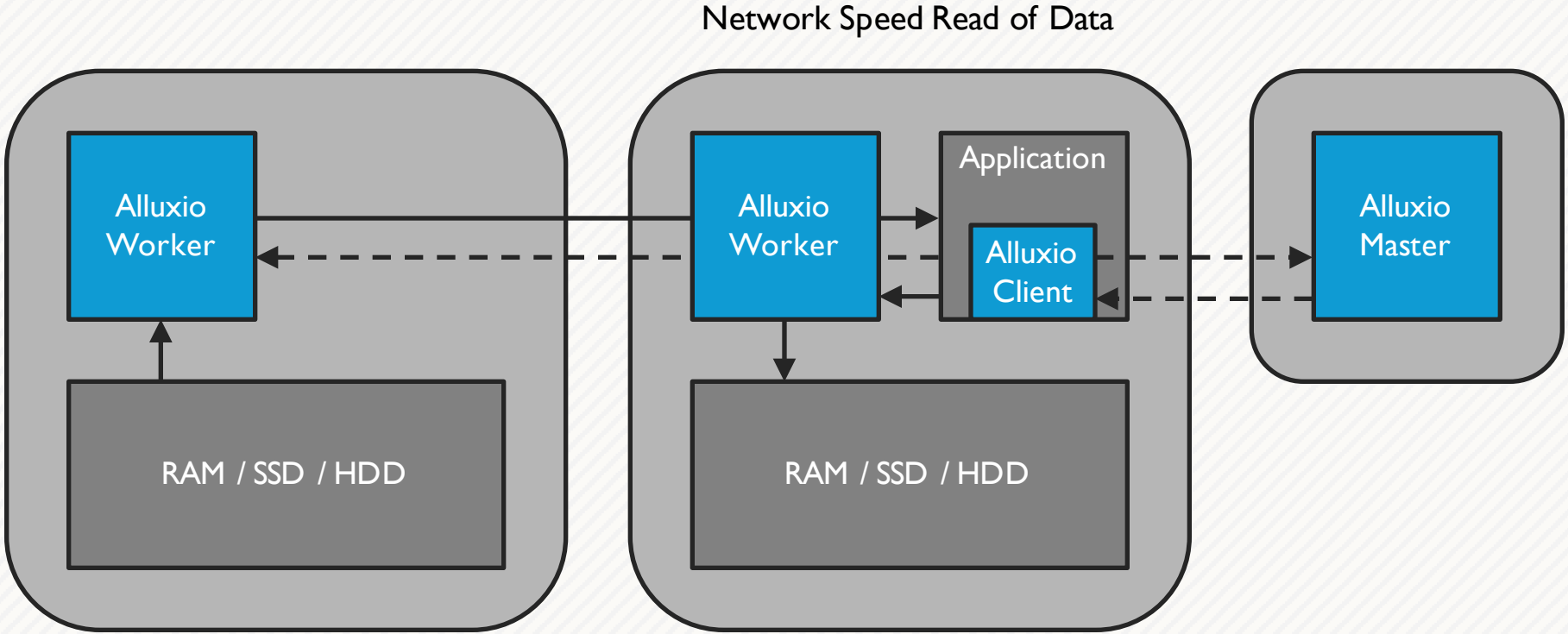
Read data in Alluxio, on same node as client



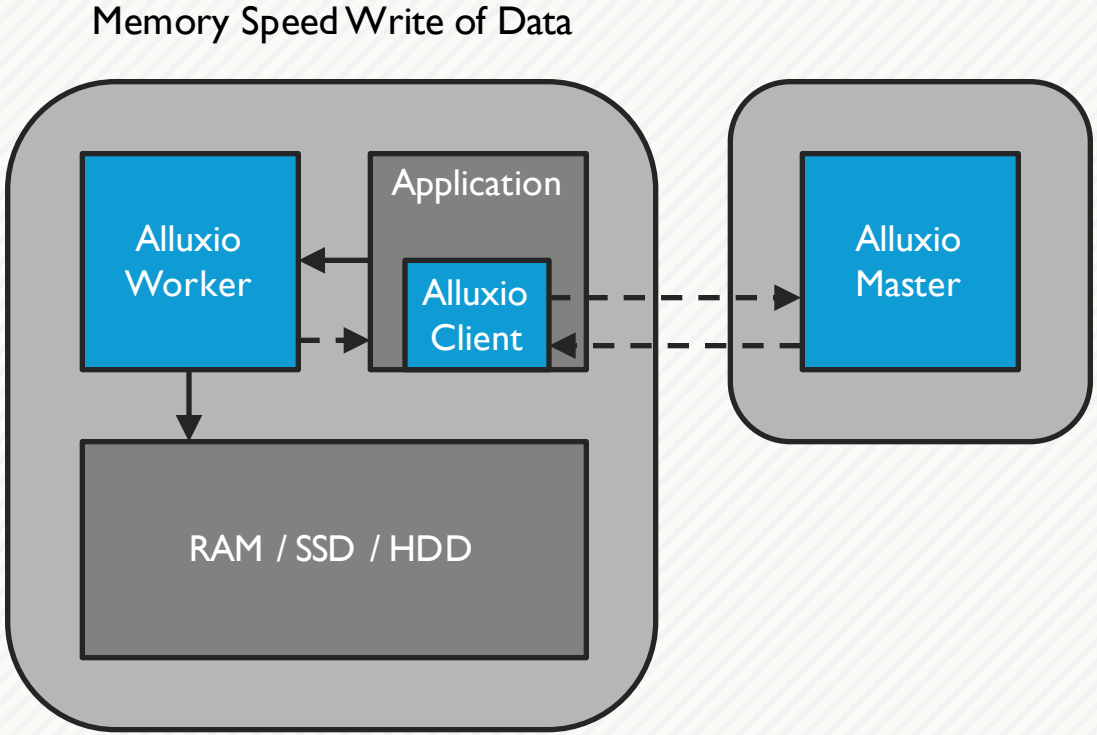
Read data not in Alluxio + Caching



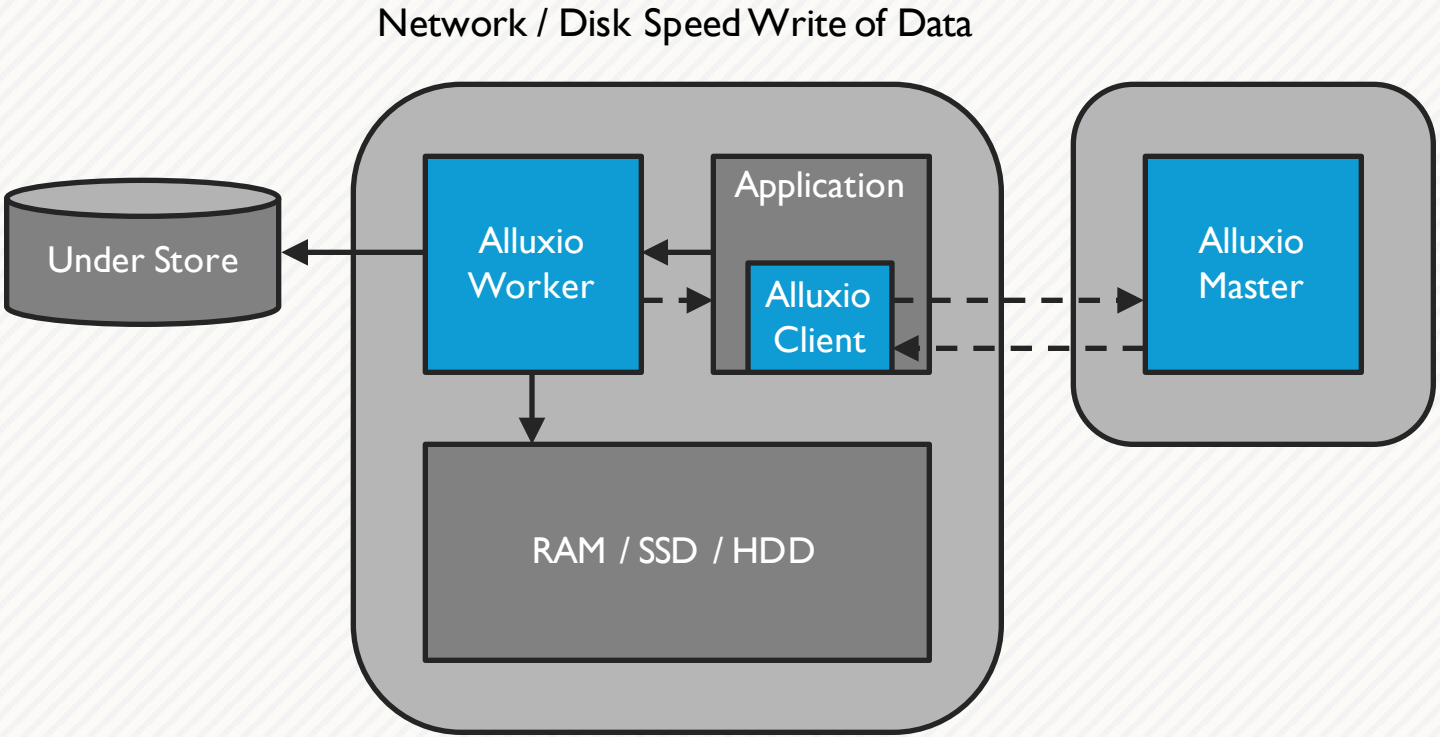
Read data in Alluxio, not on same node as client + Caching



Write data only to Alluxio



Write to Alluxio and Under Store Synchronously



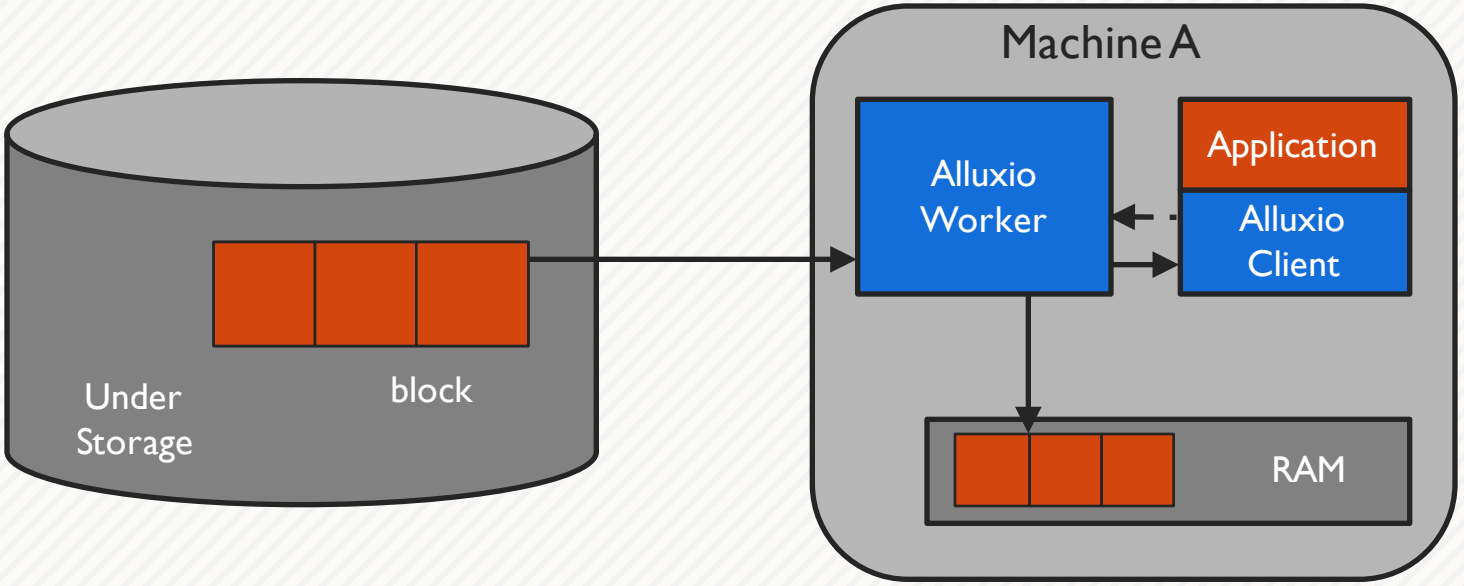
Outline

1. Why we built Alluxio
2. Alluxio's innovations
3. Alluxio's Architecture
4. What's new in 1.7

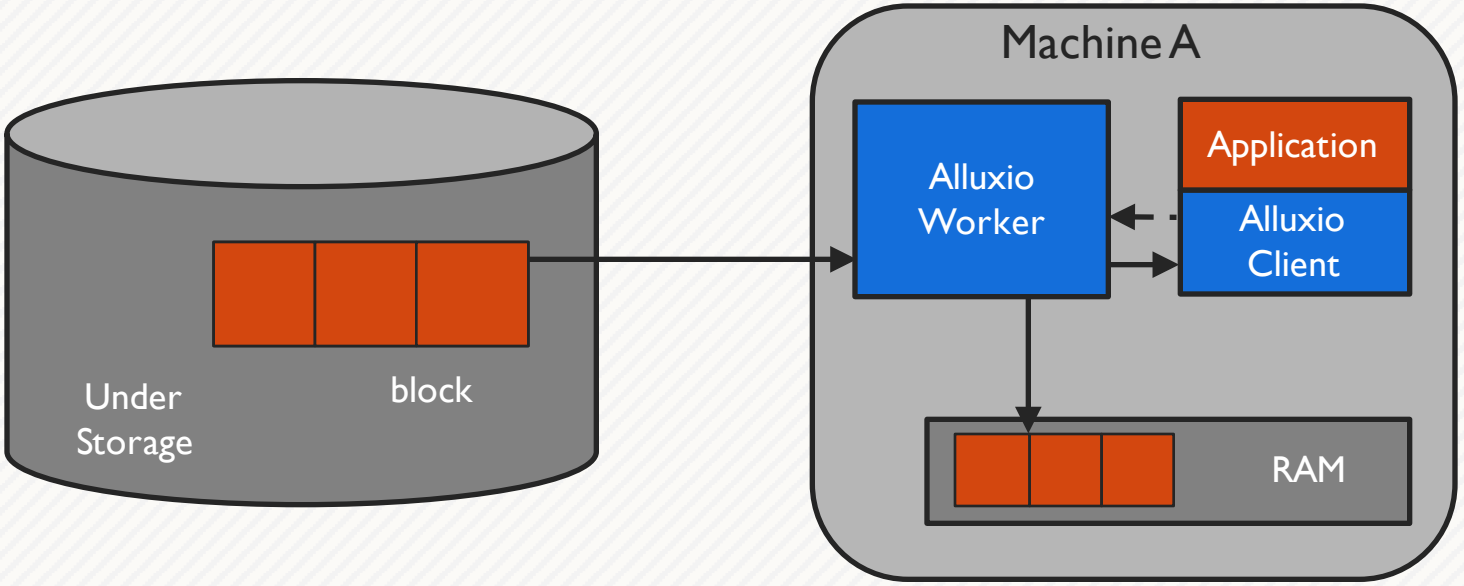
New features in 1.7

- Async caching
- Kubernates integration
- Tiered locality
- Under store synchronization
- FUSE improvement

Partial Caching



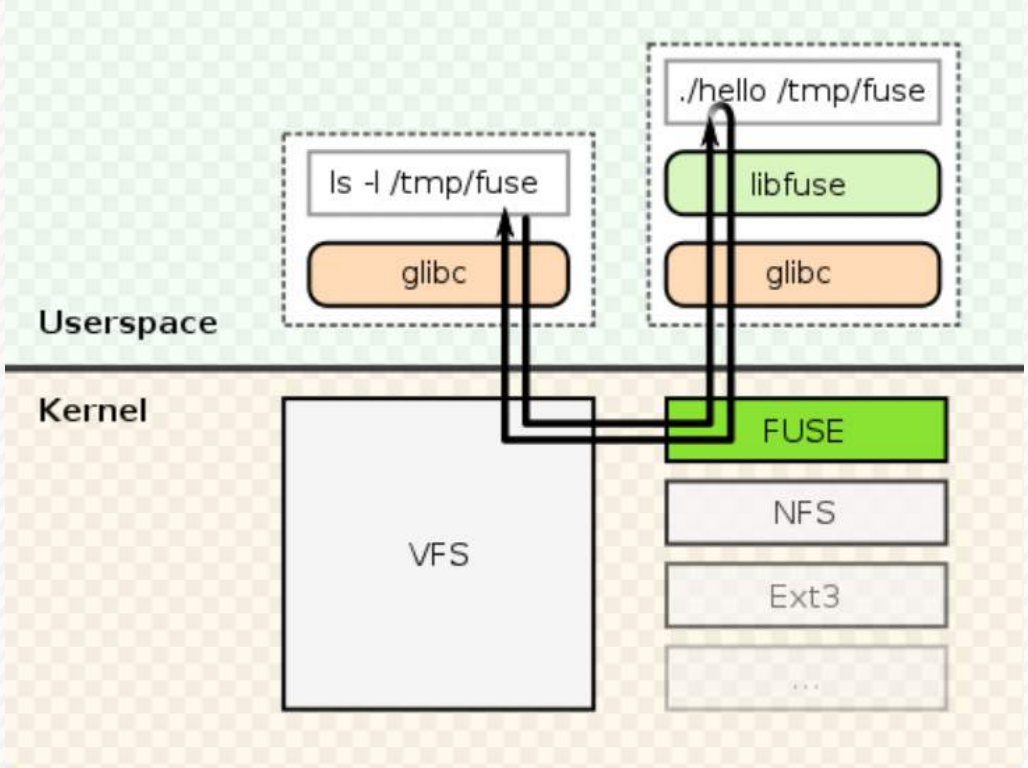
Async Caching



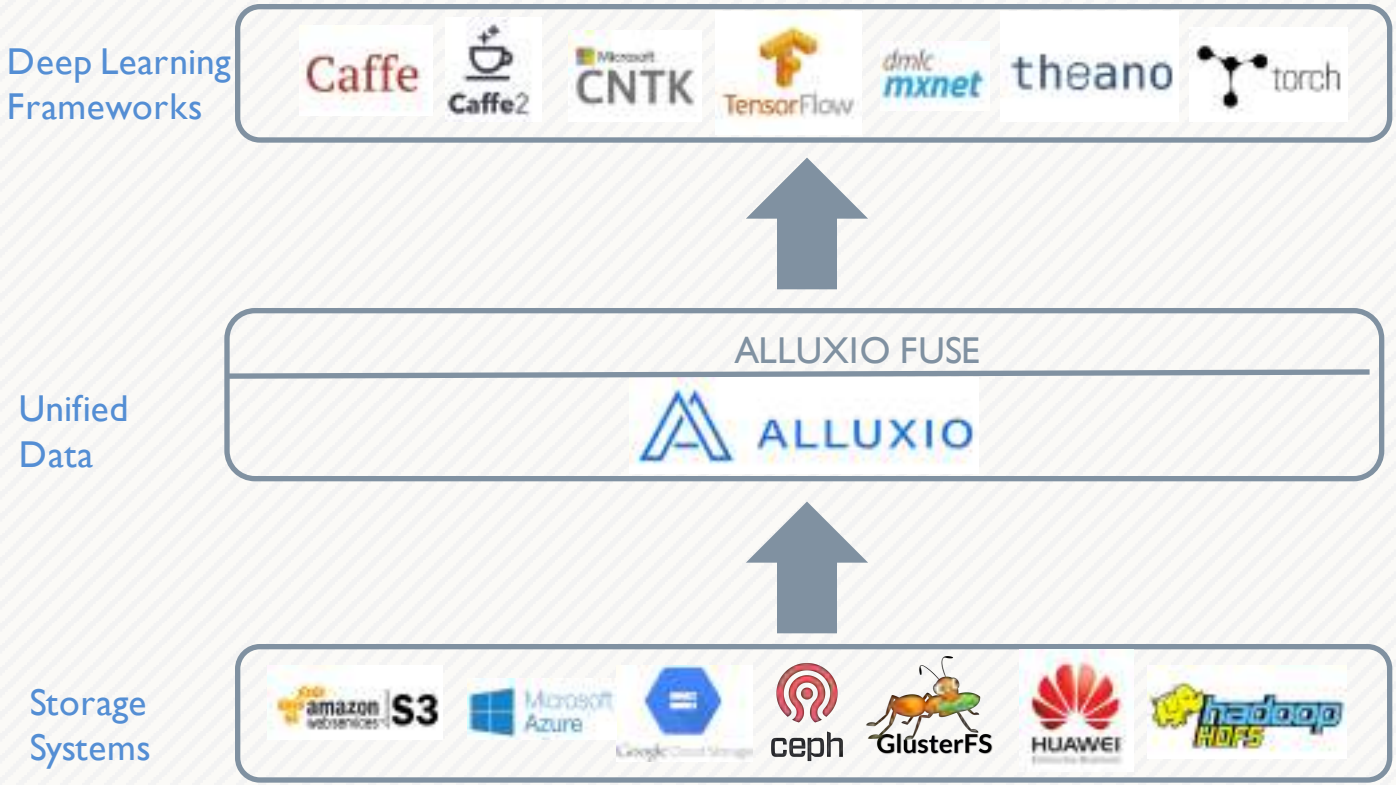
Filesystem in Userspace (FUSE)

Running file system code

in user space



Alluxio FUSE

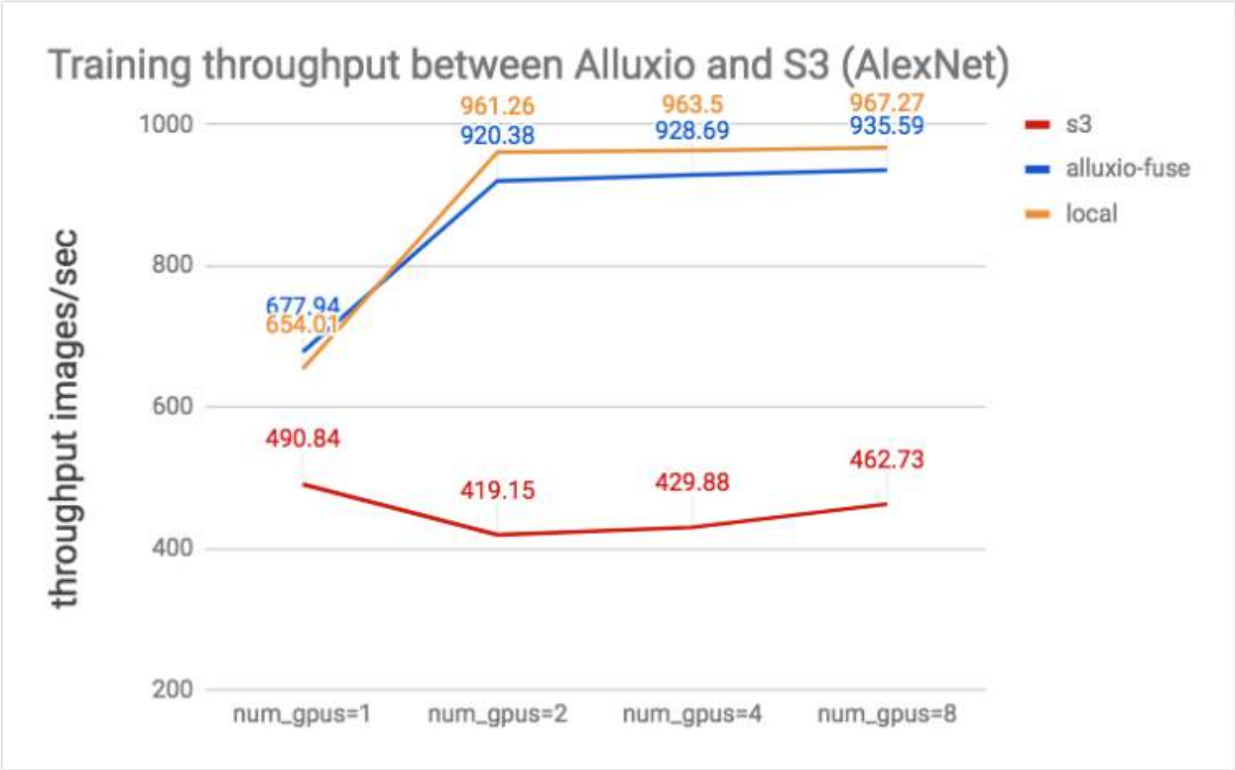


Deep Learning Input Pipeline

Deep Learning training involves three stages of utilizing different resources:

- Data reads (I/O): e.g. choose and read image files from source.
- Data Preprocessing (CPU): e.g. decode image records into images, preprocess, and organize into mini-batches.
- Modeling training (GPU): Calculate and update the parameters in the multiple convolutional layers

Alluxio overcomes I/O bottleneck



Thank you!

Bin Fan
binfan@alluxio.com
Github:apc999

知乎专栏　zhuanlan.zhihu.com/alluxio

微信公众号

Website

www.alluxio.org

www.alluxio.com

E-mail

info@alluxio.com

Social Media

Twitter.com/alluxio

Linkedin.com/alluxio

极客邦科技
企业培训与咨询

Geekbang.

扫码关注

获取更多培训信息