Second Workshop on Architectures and Systems for Big Data (ASBD 2012)

June 9th, 2012

HFAA: A Generic Socket API for Hadoop File Systems

Adam Yee and **Jeffrey Shafer**
University of the Pacific

Hadoop MapReduce

- **Hadoop:** Open source framework for data-intensive computing
  - Inspired by Google's web indexing framework
  - Uses **MapReduce** parallel programming model
    - Enables scalable computation on a commodity **cluster computer**
  - Popular and in widespread use today
    - Amazon, Facebook, Microsoft Bing, Yahoo, …

Hadoop Software Stack

- Hadoop is an all-in-one **software framework that** ties the cluster together
  - **Computation** – Execute Map and Reduce tasks
  - **Storage** – User-level filesystem for applications
    - **HDFS – Hadoop Distributed File System**
  - **Scheduling** – Distribute jobs across cluster
  - **Reliability** – Data replication, re-spawning failed jobs
- Designed for **portability** (Written in **Java**)

Motivating Challenge

1. I already have a cluster computer
2. I already have a different distributed file system running (and expertise in managing it)
   - File system examples: *PVFS*, *Ceph*, *Lustre*, *GPFS*
   - Will refer to them collectively as *NewDFS* for the remainder of talk
3. Hadoop (MapReduce) is only a small part of my computation workload

Hadoop needs to come to me, and not the other way around…

Motivating Challenge

- This is **harder than it sounds**!
- **Hadoop is tightly integrated with *HDFS***
  (Hadoop Distributed File System)
  - Holds data input and computation output

How can I use Hadoop to process data stored in other distributed file systems?

Use Hadoop with *NewDFS*

- **Method 1:** Copy the data from *NewDFS* to *HDFS*
  - Pros: Easy ☺
  - Cons: **Slow** and wastes storage space
- **Method 2:** Mount *NewDFS* using a POSIX driver (which can be directly accessed in Hadoop)
  - Pros: Easy; Faster than making a copy first! (but still slow)
  - Cons: Lose Hadoop performance optimizations (like data locality)
- **Method 3:** Custom software layer integrates directly with Hadoop
  - Pros: Near-native speed
  - Cons: Highest complexity; Requires detailed knowledge of Hadoop and *NewDFS* architecture

Using Hadoop with *NewDFS*

Past Projects

- Hadoop with **CloudStore**
- Hadoop with **Ceph**
- Hadoop with **GPFS**
- Hadoop with **Lustre**
- Hadoop with **PVFS**
- **So is this a solved problem?**
  - **No!**

Limitations of Prior Work

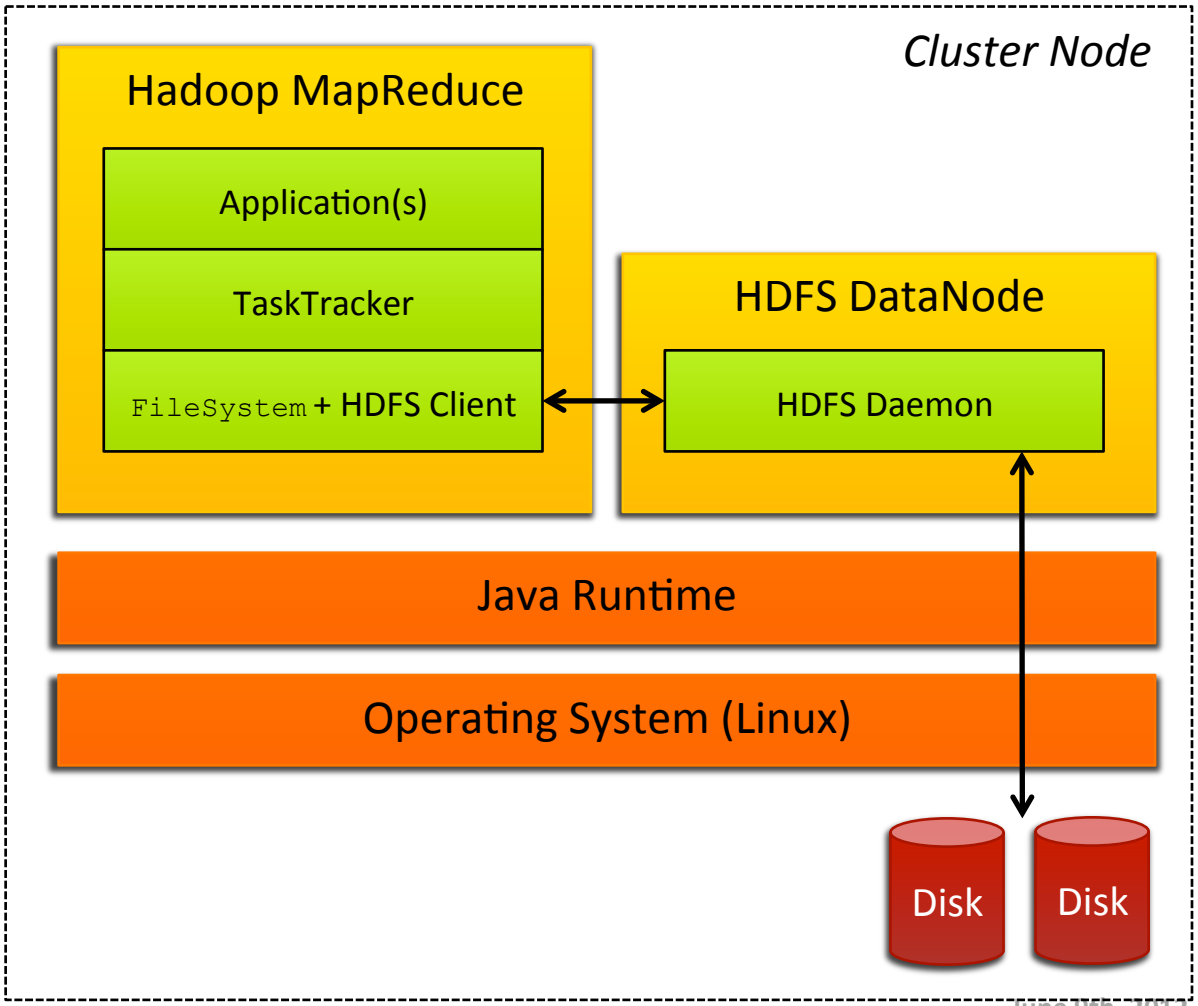
- All of these are point solutions
  - Different implementation strategies
  - Each require different software patches to Hadoop

Hadoop Filesystem Agnostic API (HFAA)

- **Universal, generic interface**
- Allows Hadoop to run on any file system that supports network sockets
  - *Since we're targeting distributed file systems, that should include everyone*
- Design moves integration responsibilities outside of Hadoop
  - Does not require user or developer knowledge of the Hadoop framework

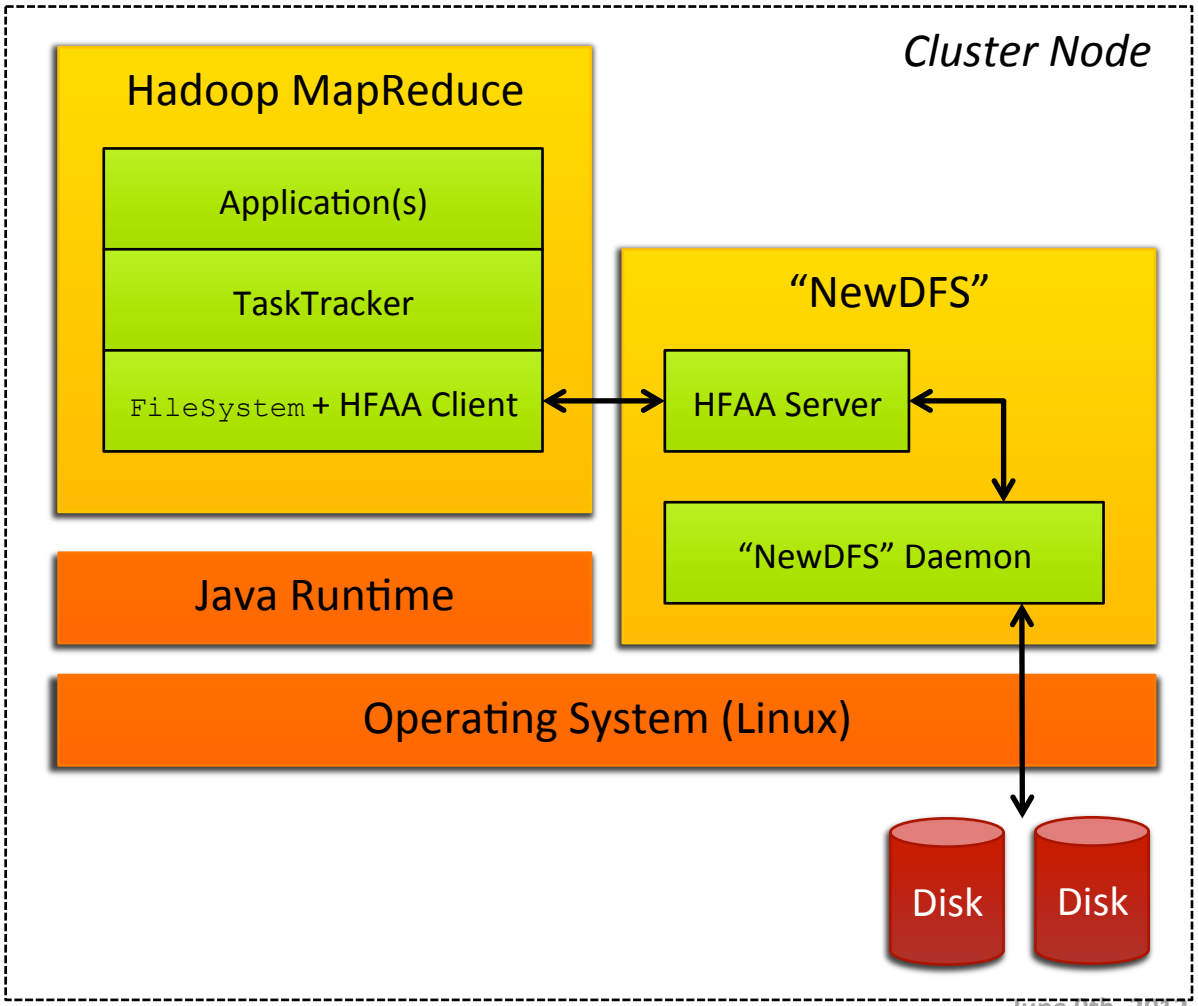
Traditional Hadoop

- `FileSystem` is Hadoop's storage API class
  - **HDFS**
  - Local disk
  - Amazon S3
- Java implementation



Hadoop + HFAA

- **Client:**
  Integrates with Hadoop (Java)
  - Reusable for any *NewDFS*
- **Server:**
  Integrates with *NewDFS* (Any language)



HFAA Operations

- Fundamental Hadoop storage operations
- How do we know API is complete?
  - Every class that extends `FileSystem` (including HFAA) must implement these abstract methods

Function
Open
Create
Append
Rename
Delete
List Status
Make Directories
Get File State
Write
Read

Evaluation

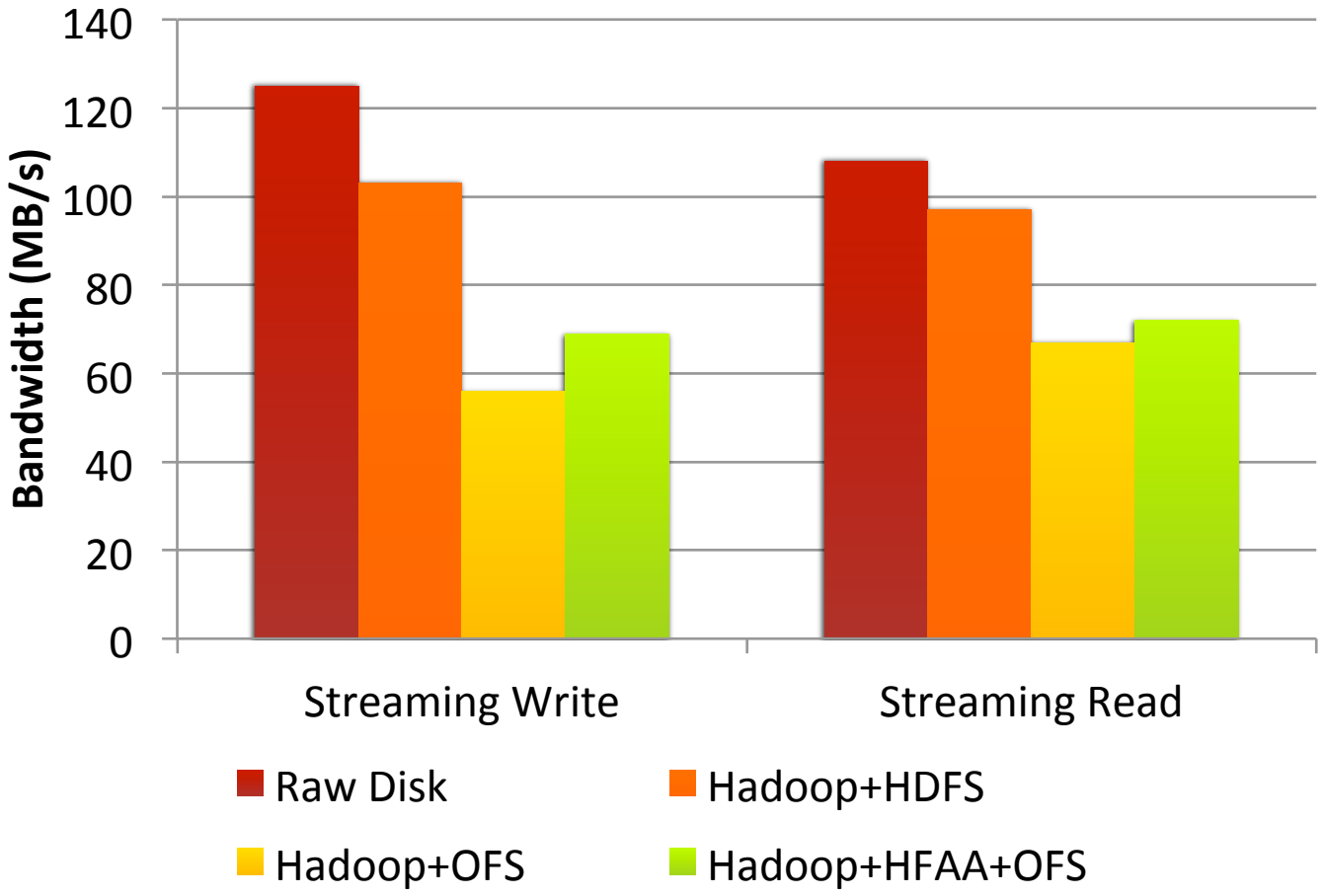
System

- HFAA prototype written for Hadoop + PVFS
  - Hadoop 0.20.204.0
    - Released Sept 5th, 2011
  - OrangeFS 2.8.4
    - Branch of PVFS
- Run on small 4-node cluster
- Streaming reads and writes

Architectures Compared

1. Raw disk (outside Hadoop)
2. Hadoop with native HDFS
3. Hadoop with OrangeFS using POSIX driver
4. Hadoop with OrangeFS using HFAA

Evaluation


140
120
100
80
60
40
20
0
Bandwidth (MB/s)
Streaming Write
Streaming Read
Raw Disk
Hadoop+HDFS
Hadoop+OFS
Hadoop+HFAA+OFS

Next Steps

Performance Optimizations

- Expose **file system locality** to Hadoop scheduler
  - More important when we test on larger clusters
- **Socket re-use** between requests
  - Less detrimental because HDFS moves 64MB data block per request
- **Tune**, **tune**, **tune**!

Broader Compatibility

- Implement HFAA Server component for **other popular file systems**
  - Lustre?
  - Ceph?
  - Others?
- **Release** for **Hadoop 1.0.x** family

Summary

- **Hadoop Filesystem Agnostic API (HFAA)**
  - Generic interface supports any distributed file system
- Implementation includes two components
  - HFAA Client – Interfaces with Hadoop
  - HFAA Server – Interfaces with PVFS
- Client can be re-used with all future filesystems
  - *Have a new filesystem you like?*
  - You only need to understand your filesystem and our simple API to link it to Hadoop
    - You don't have to be a Hadoop expert

Questions?