Evaluating BlobSeer for Map/Reduce applications

Diana Moise

diana.moise@inria.fr

1. Map/Reduce
    - Hadoop Map/Reduce
    - Hadoop Core
2. Specialized File Systems for MR Applications
    - Dedicated Map/Reduce File Systems
    - Hadoop Distributed File System
3. BlobSeer as storage for Hadoop
    - Integrating BlobSeer with Hadoop
    - Experimental evaluation
        - Microbenchmarks
        - Experiments with Map/Reduce Applications
4. Improving Hadoop
    - Supporting append in Hadoop
    - Intermediate data management in Hadoop
5. Conclusions

What is Map/Reduce?

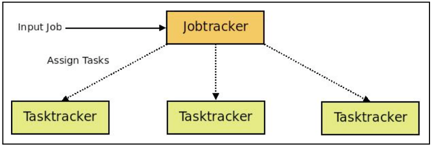
- Parallel programming model for large clusters
- Processes large amounts of data
- Provides a clean abstraction for the programmer
  - Communication between nodes
  - Parallelization (scheduling and data distribution)
  - Fault tolerance

Hadoop's Map/Reduce implementation

- Open-source Java project
- Large scale computation and data processing
- Works on comodity hardware
- Founded by Apache
- In production use at Yahoo, Facebook, Amazon, IBM...

Hadoop Core

- Hadoop Distributed File Systems (HDFS)
- Hadoop MR framework
  - Jobtracker
    - acts as master
    - splits input data
    - schedules tasks
    - monitors and re-executes the failed tasks
  - Tasktrackers
    - act as slaves
    - execute map and reduce tasks



Dedicated File Systems for M/R

1. GoogleFS
   - Chunkservers store files split into 64MB chunks
   - Centralized master server
     - keeps metadata about directory structure and chunk location

Dedicated File Systems for M/R

1. GoogleFS
   - Chunkservers store files split into 64MB chunks
   - Centralized master server
     - keeps metadata about directory structure and chunk location
2. Amazon's Map/Reduce - Elastic MapReduce
   - Map/Reduce as a service
   - Hadoop on Elastic Compute Cloud (EC2)
   - Storage backend - Simple Storage Service (S3)

Dedicated File Systems for M/R

1. GoogleFS
   - Chunkservers store files split into 64MB chunks
   - Centralized master server
     - keeps metadata about directory structure and chunk location
2. Amazon's Map/Reduce - Elastic MapReduce
   - Map/Reduce as a service
   - Hadoop on Elastic Compute Cloud (EC2)
   - Storage backend - Simple Storage Service (S3)
3. File Systems for HPC adapted for Map/Reduce
   - IBM's GPFS (General Parallel File System)
   - PVFS (Parallel Virtual File System)

Hadoop Distributed File System

- Follows GFS's model
- Two server types:

  Namenode - keeps the metadata

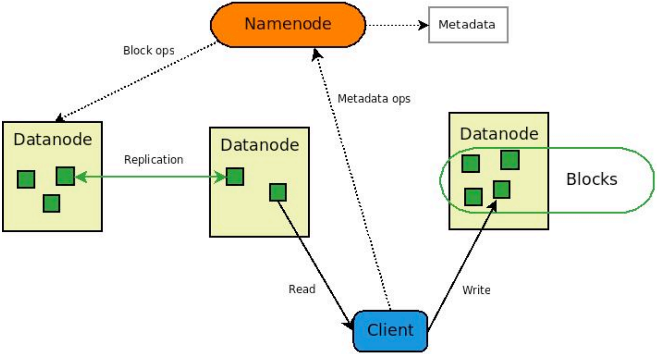
  Datanode - stores the data
- Failures handled through block level replication

  3 replicas kept: locally, in the same rack, on a different rack
- Only one writer at a time, no overwrites, no appends
- Optimizations:

  client-side buffering for small I/O ops (usually 4KB)

  exposes the mapping of block to datanodes

Hadoop Distributed File System - Architecture



Integrating BlobSeer with Hadoop

- Java API

  basic file system operations: create, read, write...
- BlobSeer File System (BSFS)
  - File system namespace - keeps file metadata, maps files to BLOB's
  - Data prefetching
  - Exposing data distribution

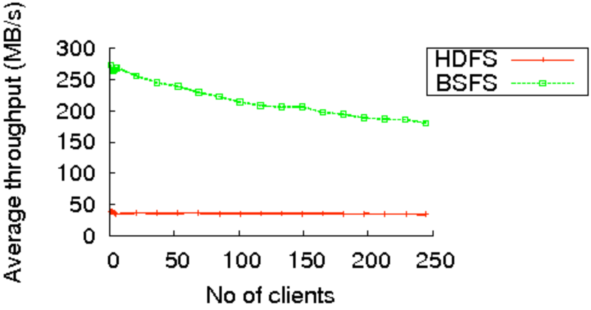
Testing and evaluation - overview and goals

- Goal
  - Measure the throughput of HDFS and BSFS under different scenarios
  - Evaluate the impact of replacing HDFS with BSFS
- Test scenarios
  - Microbenchmarks
    - Direct access to the file system
    - Common access patterns in Map/Reduce applications
  - Real Map/Reduce Applications
    - Distributed grep
    - Distributed sort

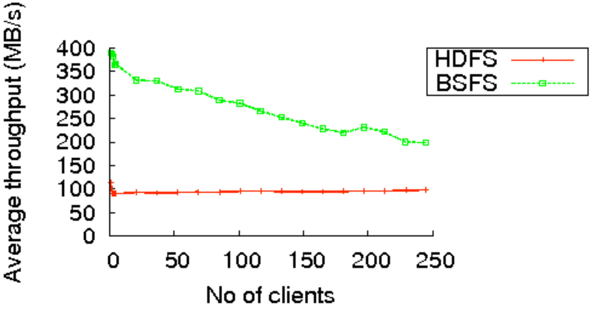
Microbenchmarks - setup

- Microbenchmarks
  - 270 nodes from the same cluster on Grid'5000
  - HDFS:
    - one namenode on a dedicated machine
    - one datanode on each cluster node
  - BSFS:
    - one vmanager, one pmanager, one namespace manager
    - 20 metadata providers
    - providers on the rest of the nodes

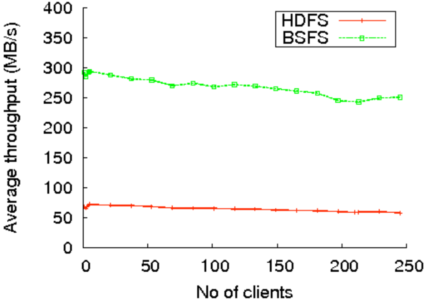
Scenario 1: concurrent clients writing to different files



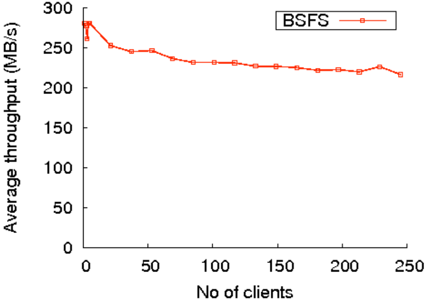
Scenario 2: concurrent clients reading from different files



Scenario 3: concurrent clients reading different parts from the same file

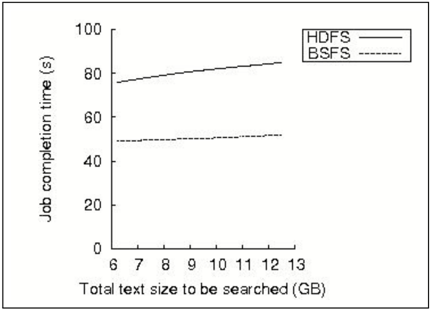


Scenario 4: concurrent clients appending data to the same file



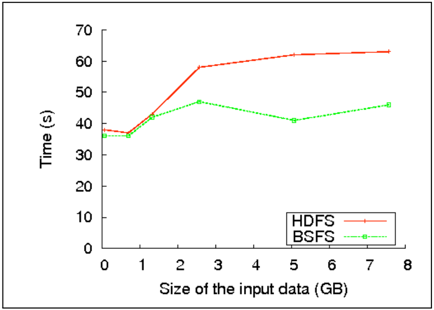
Distributed grep

- A distributed job, huge input data
- Scan a huge text file for occurences of a particular expression
- Output the number of occurences



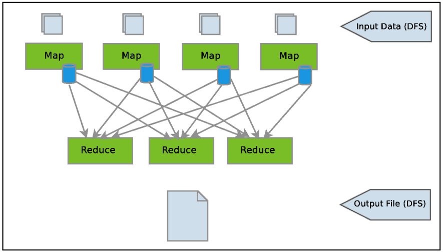
Distributed sort

- Sorts key-value pairs
- Both read and write intensive

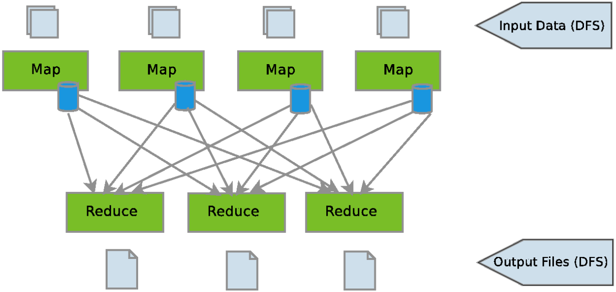


Supporting append in Hadoop

- Append implemented at the file system level
- Modify reducer code in Hadoop to append the output to a single file
- Improve execution time for pipeline MapReduce applications
- Reduce metadata associated to files



Intermediate data management in Hadoop



- Original Hadoop

  in case of mapper failure, the data is lost

- Approach

  store the intermediate data in the DFS

- Master project (work in progress)

Conclusions

- BSFS improves Hadoop's throughput
- Hadoop can be extended/improved by using BSFS's features:
  - concurrent appends
  - concurrent writes at random offsets
  - versioning
- Future work
  - pipeline MapReduce applications
  - intermediate data management
  - MapReduce in Clouds