Spark and S3

Ryan Blue
Spark Summit 2017

NETFLIX

Contents.

- Big Data at Netflix.
- HDFS, S3, and rename.
- Output committers.
- S3 multipart committer.
- Results & future work.

Big Data at Netflix.

Big Data at Netflix.

500B to 1T

daily events

60+ PB

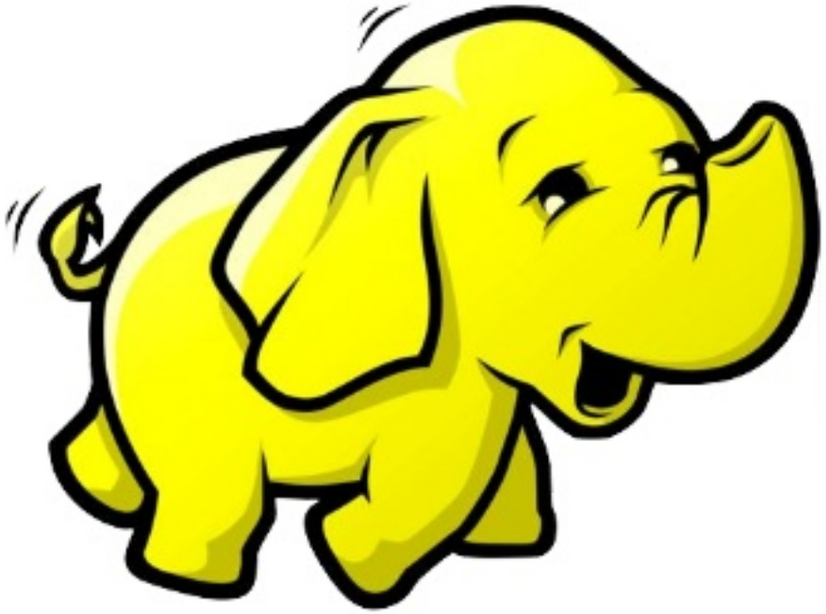
data warehouse

5 PB

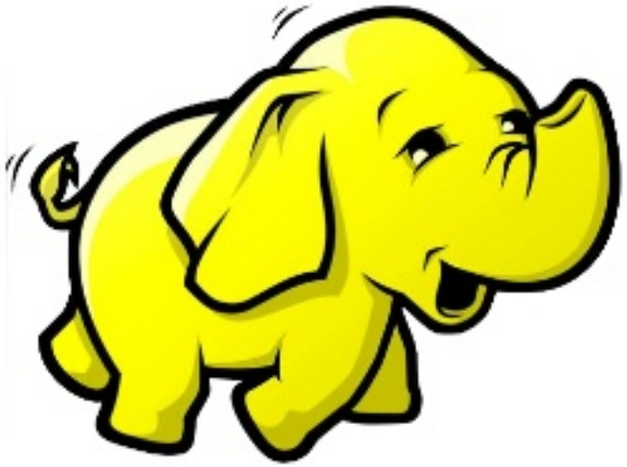
read daily

300 TB

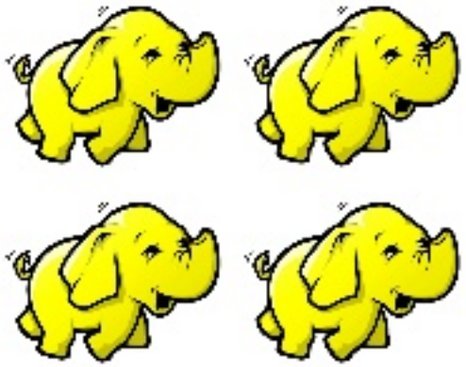
written daily



Production
3400 d2.4xl
355 TB memory



Ad-hoc
1200 d2.4xl
125 TB memory



Other clusters

Netflix clusters are expendable.

- Need to update YARN? Deploy a new cluster.
- Reconfigure NodeManagers? Deploy a new cluster
- Add temporary capacity? Deploy a new cluster.
- Lost the NameNode? Deploy a new cluster. ***Quickly***.

Expendable clusters require architectural changes.

- **GENIE** is a job submission service that selects clusters
- **METACAT** is a cluster-independent metastore
- **S3** is used for all data storage

S3 != HDFS.

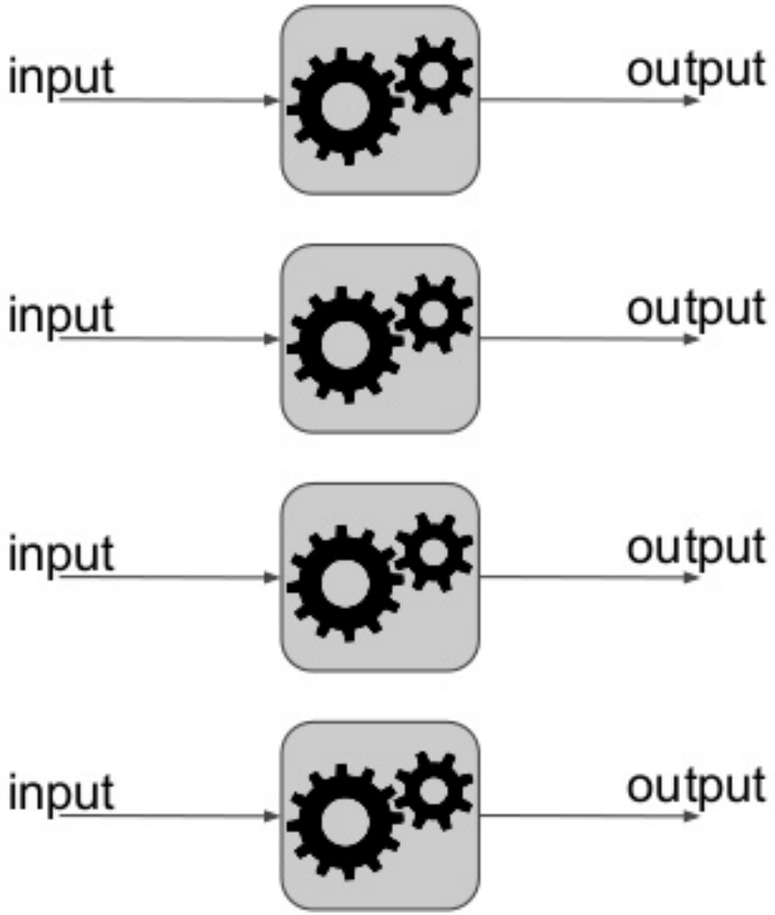
- A distributed object store (masquerading as FileSystem).
- Rename results in a copy and delete.
- File output committers **rename every output file**.

Why Output Committers?

Why commit outputs?

- DirectOutputCommitter writes directly to the final location.

We tend to think of tasks like this:



But Spark is a distributed system.

So reality is closer to this:

Anything can happen.

- Spark might lose communication with an executor.
- YARN might preempt executors.
- Speculative execution may start duplicate tasks.
- Tasks may run slowly, but still try to commit.

In practice, execution is this:

Committers clean up the mess.

- Task attempts write different files in parallel.
- One attempt per task commits.
- Once all tasks commit, the job is commits.

Committer guarantees*:

- One (and only one) copy of each task's output.
- All output from a job is available, or none is.

Committer guarantees*:

- One (and only one) copy of each task's output.
- All output from a job is available, or none is.

*Not really guarantees.

Why commit outputs?

- DirectOutputCommitter writes directly to the final location.
  - Concurrent attempts clobber one another.
  - Job failures leave partial outputs behind.
  - Removed in Spark 2.0.0 – SPARK-10063

S3MultipartOutputCommitter.

A simplified file committer.

- **Attempts**: write to a unique file for each task/attempt.
- **Task commit**: rename attempt file to task file location.
  - `mv /tmp/task-4-attempt-0 /tmp/job-1/task-4`
- **Job commit**: rename job output to final location.
  - `mv /tmp/job-1 /final/path/output`

- This is (roughly) what FileOutputCommitter does.
- Move/rename is not a metadata operation in S3!

S3 multipart uploads.

- Incremental file upload API.
- Upload file blocks as available.
- Notify S3 when finished adding blocks.

S3 multipart uploads.

- Incremental file upload API.
- Upload file blocks as available.
- **Notify S3 when finished adding blocks.**

Multipart upload committer.

- **Attempts**: write to a unique file on local disk.
- **Task commit**:
  - Upload data to S3 using the multipart API.
  - Serialize the final request and commit that to HDFS.
- **Job commit**:
  - Read the task outputs to get final requests
  - Use the pending requests to notify S3 the files are finished

Failure cases.

- Will abort uploads to roll back.
- Will delete files to clean up partial commits.
- Failures during job commit can leave extra data.

  (but it's no worse than the file committer.)

Results.

- Metadata-only job commit (unlike file committer).
- Provides reliable writes to S3 (unlike direct committer).
- Distributed content uploads, light-weight job commit.
- Shortened job commit times by **hours**.

S3 multipart committer

- Released single-directory and partitioned committers.
  - `https://github.com/rdblue/s3committer`
- Will be included in S3A – HADOOP-13786
  - Huge thanks to Steve Loughran!

Future work.

- Short term, finish integrating into Hadoop.
- Long term, Hive table layout needs to be replaced.

Thank you!
Questions?

NETFLIX

https://jobs.netflix.com/
rblue@netflix.com