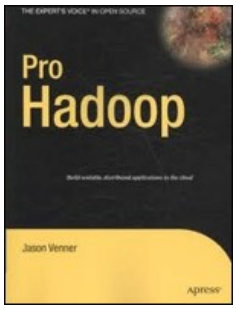

Pro Hadoop

by Jason Venner

Chapter 10: Projects Based On Hadoop and Future Directions

People use Hadoop to solve many types of problems, and a number of teams have built packages on top of Hadoop Core to address an even larger scope of problems. This chapter will walk through some of the many tools being built on top of Hadoop and one tool that can be built into Hadoop. Hadoop Core is an evolving project: over the time of writing this edition of the book, Hadoop 0.19.0 and Hadoop 0.19.1 came out, and Hadoop 0.20.0 became available in May 2009. (You'll see a section on changes later in this chapter.)

Hadoop Core-Related Projects

The main web site for Hadoop Core, `http://hadoop.apache.org/core`, provides a list of related projects and subprojects: HBase, Hive, Pig, Mahout, and Hama. The top-level Hadoop project, `http://hadoop.apache.org/`, also includes ZooKeeper. This section will provide an overview of them and, when feasible, show a quick example of how to set up and use them (as well as what problems users might encounter).

Disclaimer

I have little to no experience with most of the projects listed in this chapter, so the information in this chapter is gleaned from reading the project or company web site and/or trying the examples from a current release.

HBase: HDFS-Based Column-Oriented Table

The project description describes HBase as the Hadoop database an open source, column-oriented structured datastore based on the Google BigTable paper, `http://labs.google.com/papers/bigtable.html`. The earlier versions of HBase used the Hadoop MapFile as the underlying storage mechanism and managed updates by maintaining overlay MapFiles. When there were sufficient updates, a merged file was reconstructed, and the overlays were discarded. To speed access and distribute access, each individual MapFile is responsible for only a specific range of data in a table column, and if the MapFile grows past a specified size, it is split into multiple MapFiles. More recent versions of HBase also provide a memcached-based intermediate layer between the user and the MapFiles (`http://www.danga.com/memcached/`).

Prior to the addition of the memcached layer, HBase suffered terrible performance for random reads and writes, primarily because HDFS is not optimized for low latency random access. Ordered reads and writes perform at near-HDFS speed.

HBase has a number of server processes, a single HBaseMaster that manages the HBase cluster and a set of HRegionServers, each of which is responsible for a set of MapFiles containing column regions.

HBase suffers terribly from the inability of applications to flush file data to storage before the file is closed, and a crash of any portion of the HBase servers or a service interrupting crash of HDFS will result in data loss.

In prior chapters there was a discussion of problems caused by applications or server processes attempting to exceed the system-imposed limit on the number of open files; HBase also has this problem. The problem is substantially aggravated because each Hadoop MapFile is actually two files and a directory in HDFS, and each HDFS file also has a hidden checksum file. Setting the per-process open file count very large is a necessity for the HBase servers. A storage file format, HFile, is under development and due for Hbase version 0.20.0, and is expected to solve many of the performance and reliability issues.

HBase relies utterly on a smoothly performing HDFS for its operation; any stalls or DataNode instability will show up as HBase errors. There are HDFS tuning parameters suggested in the troubleshooting section on the HBase wiki: `http://wiki.apache.org/hadoop/Hbase/Troubleshooting`. In particular, if the underlying HDFS cluster is experiencing a slow block report problem, `https://issues.apache.org/jira/browse/HADOOP-4584`, HBase is not recommended.

HBase servers, particularly the version using memcached, are memory intensive and generally require at least a gigabyte of real memory per server; any paging will drastically affect performance. Java Virtual Machine (JVM) garbage collection thread stalls are also causing HBase failures.

HBase generally provides downloadable release bundles that track the Hadoop Core distributions. HBase is not part of the Hadoop Core distribution.

Hive: The Data Warehouse that Facebook Built

Hive provides a rich set of tools in multiple languages to perform SQL-like data analysis on data stored in HDFS. The wonderful people at Facebook have contributed Hive to the Apache project. As of the publication of this book, Hive is undergoing active development. Compiled versions of Hive are part of the contrib subtree of the Hadoop Core distribution.

Cloudera, discussed later in this chapter, provides online training for Hive.

Setting Up and Running Hive

The following four lines are required before attempting to start Hive (your installation might already have the `/tmp` and `/user/hive/warehouse` directories present):

```
hadoop fs -mkdir       /tmp
hadoop fs -mkdir       /user/hive/warehouse
hadoop fs -chmod g+w   /tmp
hadoop fs -chmod g+w   /user/hive/warehouse
```

The only issue I encountered when running Hive was a problem with a missing JAR because of an error I introduced into the `conf/hadoop-env.sh` file (see Listing 10-1).

Listing 10-1: Hive Configuration Error

```
jason@cloud9:~/src/hadoop-0.19/contrib/hive$ bin/hive
```

```
java.lang.NoClassDefFoundError: org/apache/hadoop/hive/conf/HiveConf
    at java.lang.Class.forName0(Native Method)
    at java.lang.Class.forName(Class.java:247)
    at org.apache.hadoop.util.RunJar.main(RunJar.java:158)
    at org.apache.hadoop.mapred.JobShell.run(JobShell.java:54)
    at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:65)
    at org.apache.hadoop.util.ToolRunner.run(ToolRunner.java:79)
    at org.apache.hadoop.mapred.JobShell.main(JobShell.java:68)
Caused by: java.lang.ClassNotFoundException: org.apache.hadoop.hive.conf.HiveConf
    at java.net.URLClassLoader$1.run(URLClassLoader.java:200)
    at java.security.AccessController.doPrivileged(Native Method)
    at java.net.URLClassLoader.findClass(URLClassLoader.java:188)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:306)
    at java.lang.ClassLoader.loadClass(ClassLoader.java:251)
    at java.lang.ClassLoader.loadClassInternal(ClassLoader.java:319)
    ... 7 more
```

I modified the `conf/hadoop-env.sh` file to set the `HADOOP_CLASSPATH` (see Listing 10-2) when I was testing the scheduler services in Chapter 8. The `contrib/hive/bin/hive` script sets `HADOOP_CLASSPATH` with the set of JARs that Hive requires and then invokes the `bin/hadoop` script to start the Hive command-line interpreter.

Listing 10-2: Incorrect Modification of the HADOOP_CLASSPATH Setting in conf/hadoop-env.sh

```
# Extra Java CLASSPATH elements.  Optional.
export HADOOP_CLASSPATH=${HADOOP_HOME}/contrib/capacity-scheduler/➥
hadoop-0.19-capacity-scheduler.jar
```

I corrected the error (see Listing 10-3), and Hive started correctly (see Listing 10-4).

Listing 10-3: Corrected Setting for HADOOP_CLASSPATH in conf/hadoop-env.sh

```
# Extra Java CLASSPATH elements.  Optional.
export HADOOP_CLASSPATH=${HADOOP_HOME}/contrib/➥
capacity-scheduler/hadoop-0.19-capacity-scheduler.jar:${HADOOP_CLASSPATH}
```

Listing 10-4: Hive Starts Correctly After Constructing the Required HDFS Path Elements with the Correct Permissions

```
jason@cloud9:~/src/hadoop-0.19/contrib/hive$ bin/hive
```

```
hive>
```

The examples listed in the wiki page http://wiki.apache.org/hadoop/Hive/GettingStarted did not work particularly well for me (they might be updated by the time you read this chapter).

Pig, the Other Latin: A Scripting Language for Dataset Analysis

Pig provides a high-level language for writing SQL-like operations that apply to datasets. The language is named Pig Latin, and the Pig project provides a compiler that produces MapReduce jobs from a Pig Latin script. Pig is not distributed with Hadoop Core, and is mature enough that the project has releases. At the time of writing, Pig 0.2.0 has been released. Pig also provides `grunt`, an interactive shell, for running Pig Latin commands directly. Cloudera, listed later in this chapter, provides online training for Pig.

The site http://www.apache.org/dyn/closer.cgi/hadoop/pig provides the main distribution page. At present, it appears that the stock Pig distribution requires the underlying cluster to run Hadoop 0.17.0 or Hadoop 0.18.0.

The setup is as simple as unpacking the distribution and setting the environment variable `PIG_CLASSPATH` to the directory that contains the `hadoop-site.xml` file that defines your cluster. The following should work:

```
export PIG_CLASSPATH=${HADOOP_HOME}/conf
```

Mahout: Machine Learning Algorithms

The Mahout project aims to build scalable machine learning algorithms. Its plan is to build libraries for the ten machine learning algorithms listed in http://www.cs.stanford.edu/people/ang//papers/nips06-mapreducemulticore.pdf. As of the time of writing, the first release, 0.1, has been made available for download. The Taste project (a recommendation engine) has become a part of Mahout and is included in the 0.1 release. There is a tutorial available at http://lucene.apache.org/mahout/taste.html.

Mahout requires Maven for operation, and it is not clear from the documentation how to run the examples, including the Taste examples, without Maven.

Mahout also provides a number of distributed clustering algorithms, including k-means, dirichlet, mean-shift, and canopy. There are also two Bayesian classifiers: the naive and the complementary naïve. An implementation of watchmaker is provided for building evolutionary algorithms and support for matrix and vector operations.

Hama: A Parallel Matrix Computation Framework

At the time of writing, Hama is an incubation project. It requires HBase as an underlying storage framework. The project is intended to be used for large-scale numerical analyses and data mining. The project will provide matrix-vector and matrix-matrix multiplication, linear equation solving, tools for working with graphs, data sorting, and methods of finding eigenvalues and eigenvectors. The project is undergoing development and is pre-release 0.1.

ZooKeeper: A High-Performance Collaboration Service

ZooKeeper provides a framework for building high-performance collaborative services. ZooKeeper maintains a shared namespace that looks very similar to a hierarchical file system. Applications rendezvous on entries in the namespace. Each of these namespace entries may have data associated with it. The entry data is accessed atomically, and changes are ordered. In addition, ZooKeeper provides an ephemeral node, an entry that vanishes when the service holding the entry open disconnects. The ephemeral nodes are used to establish service masters and sets of backup servers. Ephemeral nodes are used to support redundant servers with hot failover.

ZooKeeper has been designed to be very reliable and very fast in environments in which data is primarily read.

The examples at http://hadoop.apache.org/zookeeper/docs/current/recipes. html provide ZooKeeper

recipes for two-phase commit, leader election, barriers, queues, and locks.

Lucene: The Open Source Search Engine

The Lucene project, http://lucene.apache.org/java/docs/, provides the standard open source package used for search engines. The Lucene core provides the ability to take in documents in a variety of formats and build inverted indexes out of the terms found in the documents. Lucene also provides a query engine that takes incoming queries, searches the indexes, and returns the documents that match.

Hadoop Core provides a contrib package that manages indexes that are stored in HDFS: `contrib/index/hadoop-<rel>-index.jar`. The main class, `org.apache.hadoop.contrib.index.main.UpdateIndex`, is specified in the JAR. The contrib package supports distributed indexes, shards, and unified indexes.

SOLR: A Rich Set of Interfaces to Lucene

The SOLR project, http://lucene.apache.org/solr/, is a stand-alone, enterprise-grade search service built on top of Lucene. SOLR provides XML/HTTP and JSON APIs.

Katta: A Distributed Lucene Index Server

The Katta project, http://katta.sourceforge.net/, describes itself as Lucene in the Cloud, a scalable, fault-tolerant, distributed indexing system capable of serving large replicated Lucene indexes at high loads. Katta uses ZooKeeper to coordinate among the individual servers of the Katta cloud. Katta supports storing shards on the local server file system, HDFS, and in Amazon's S3. Katta also provides a distributed scoring service, allowing for the search results from multiple indexes to be merged together.

Thrift and Protocol Buffers

Thrift (http://incubator.apache.org/thrift/) and Protocol Buffers (http://code.google.com/p/protobuf/) provide a mechanism for using arbitrarily complex data types as keys or values within Hadoop. The core concept is that of defining a type in a text file and having a tool generate per-language APIs for accessing the data structure and for serializing and deserializing the data structure. As of Hadoop 0.17.0, the framework supports using any type that provides serialization services as a key or a value.

Cascading: A Map Reduce Framework for Complex Flows

Cascading, http://www.cascading.org/, describes itself as a rich API for handling complex scale-free workflows reliably on a MapReduce cluster. The Cascading package allows the rapid wiring of components together into workflows that support flow control statements. Cascading's metaphor is that the incoming data flows through a series of functions and filters that allow the data to be split into multiple streams and then joined together again as needed. An acyclic-directed graph is built by the framework, out of the functions and filters.

CloudStore: A Distributed File System

CloudStore, http://kosmosfs.sourceforge.net/ (formerly known as the Kosmos file system), provides an alternative file system for use within a MapReduce cluster. Unlike HDFS, CloudStore is implemented in C++.

Hypertable: A Distributed Column-Oriented Database

The Hypertable project, http://www.hypertable.org/, provides a distributed database conceptually similar to HBase and BigTable. The Hypertable site is clear that the project is at a 0.9 release. Currently, the core servers for Hypertable, the Master server and Hyperspace server, are single points of failure. Hypertable does not provide ready-to-run distributions and must be built from source. There are build instructions for CentOS 5.1 and CentOs 5.2, Fedora Core 8 32bit, Gentoo 2007.0, Ubunto 8.10 Intrepid Ibex 32-bit, Max OS X 10.5 Leopard, and Mac OS X 10.4 Tiger.

Hypertable provides HQL, a SQL-like language for running queries.

Greenplum: An Analytic Engine with SQL

Greenplum, http://www.greenplum.com/, provides petabyte-scale, scalable database analytics. It provides a download link to allow you to try its software. It also provides an in-database MapReduce that interoperates with SQL.

CloudBase: Data Warehousing

The CloudBase project, http://cloudbase.sourceforge.net/, provides a high-performance, data warehousing system built on top of MapReduce, with an ANSI SQL API. The project is developed by business.com to speed terabyte scale web log analysis. The current release version is 1.3. CloudBase is released under GLP 2.0. The web site provides detailed instructions for running CloudBase instances on Amazon's elastic compute (EC2) service.

Hadoop in the Cloud

Sometimes you need additional compute resources for only a short time, you want to experiment with particular configurations, or you just don't want to manage your own hardware. Cloud service vendors provide the ability to spin up clusters of almost arbitrary size and capacities for short to long durations. The best-known cloud server provider at the time of writing is Amazon, and there is direct support for running Hadoop in its cloud.

Amazon

Amazon, http://aws.amazon.com, provides a large set of cloud computing services:

- Its simple storage S3 service, http://aws.amazon.com/s3/, provides large persistent data storage.
- Its EC2 service, http://aws.amazon.com/ec2/, provides on-demand computing clusters built of virtual computers with a variety of capacities and operating systems.
- The SimpleDB, http://aws.amazon.com/simpledb/, provides a production-grade, distributed, column-oriented database.
- The Elastic Block Store (EBS), http://aws.amazon.com/ebs/, provides persistent storage within EC2 and is ideal for longer-running HDFS clusters.
- The Elastic MapReduce service provides on-demand Hadoop clusters, using S3 as the job input and output file system.

The one significant downside to Hadoop in the Amazon cloud is that there is no real data locality something Hadoop works hard to achieve.

> **Caution** Anything stored on an EC2 machine instance vanishes when the instance is shut down. Do not use EC2 instances for valuable data. Use the EBS or S3 for persistent storage.

Cloudera

Cloudera, http://www.cloudera.com/, provides a supported Hadoop distribution. At the time of writing, the base was Hadoop 0.18.3, with important fixes and features back ported from later versions, including unreleased versions. This is an ideal distribution for production use because it provides minimal API changes while providing bug fixes and some new features.

Training

Cloudera also provides a graduated series of training, from basic to advanced. It provides free online basic Hadoop training at http://www.cloudera.com/hadoop-training-basic, Hive training at http://www.cloudera.com/hadoop-training-hive-introduction, and Pig training at http://www.cloudera.com/hadoop-training-pig-introduction/. There is also a session on using Eclipse with Hadoop at http://www.cloudera.com/blog/2009/04/20/configuring- eclipse-for-hadoop-development-a-screencast/.

Supported Distribution

Cloudera provides a freely downloadable version of its distribution at http://www.cloudera.com/hadoop and a vmware image for training purposes at http://www.cloudera.com/hadoop-training-virtual-machine. The virtual machine has an Eclipse installation set up for use with its Hadoop distribution.

> **Note** I used the Cloudera training virtual machine to work up some of the examples in this book.

Cloudera also provides ready-to-use Amazon EC2 machine images (AMIs) at `http://www.cloudera.com/hadoop-ec2`. The EC2 image has Hive and Pig installed and ready to use.

Paid Support

Cloudera also provides support contracts for installations using its Hadoop distribution.

Scale Unlimited

Scale Unlimited, `http://www.scaleunlimited.com/`, provides Hadoop Core training and consulting. The principals are the Cascading project lead and the Katta project lead. From `http://www.scaleunlimited.com/consulting`:

> *Our consultants' experience does not end with Map Reduce patterns and Hadoop Distributed File System deployment models; but also spans over a wide set of related open source technologies like HBase, ZooKeeper, Cascading, Katta, Pig, Mahout, Casandra, and CouchDB.*

Scale Unlimited also sponsors a live CD image of a Solaris installation with a three-node Hadoop cluster in zones (`http://opensolaris.org/os/project/livehadoop/`).

Note A live disk is a CD or DVD that boots as a running instance, not requiring any changes to the local machine's hard disk. An image is an `.img` file that most CD/DVD burner applications can burn directly to writable media.

API Changes in Hadoop 0.20.0

Hadoop 0.20.0 introduces a number of new features and changes. At the time of writing, it is becoming clear that it is not ready for production use. This section hopes to whet your appetite for these new features and help you plan for their arrival.

Vaidya: A Rule-Based Performance Diagnostic Tool for MapReduce Jobs

Vaidya processes the log file data of previously run jobs and provides suggestions on how to improve performance.

At the time of writing, Vaidya checks the following:

- How evenly the data is partitioned between the reduce tasks
- Whether map task failure and re-executions are affecting the overall job performance
- Whether reduce task failure and re-executions are affecting the overall job performance
- Whether the `io.sort.space` size is sufficient to prevent the map tasks outputs from being spilled to disk during the map-side sort phase
- Whether substantial data, other than the key/value pairs, is being read from HDFS during the map or reduce tasks

Service Level Authorization (SLA)

The SLA package provides the access control lists for the control APIs of the various Hadoop Core servers, providing some assurance that any client connecting to a server with SLA enabled is an authorized client.

Removal of LZO Compression Codecs and the API Glue

For licensing reasons, the LZO codec interface files were removed. There are plans to bring in another LZO-like codec with a license the Apache Foundations will accept.

New MapReduce Context APIs and Deprecation of the Old Parameter Passing APIs

The core of this change is that a `Mapper` or a `Reducer Context` object is passed to the `Mapper` and `Reducer` classes, in place of the `JobConf`, to `configure()`, and the `Reporter` and `OutputCollector` to `map()` and `reduce()`. The `Mapper` and `Reducer` classes now have a `setup()`, `cleanup()`, and `run()` method in place of the `configure()` and `close()`methods.

Additional Features in the Example Code

As an aid for my development of the example code in this book, I used a number of tools. This section covers the tools I find most useful.

Zero-Configuration, Two-Node Virtual Cluster for Testing

The class `com.apress.hadoopbook.RunVirtualCluster` in `test/src` of the examples will start and run a mini-Hadoop cluster that provides a near-full Hadoop Core installation. This is ideal for use when developing and testing MapReduce jobs that need more than a single reduce task and therefore cannot be run using the `local` JobTracker.

To run it, change to a directory that will be used as the virtual cluster local storage, and run the following:

```
java -jar hadooppro.jar com.apress.hadoopbook.RunVirtualCluster➥
saved_configuration.xml
```

The cluster will be started, information about the web GUI URLs will be printed to stdout, and a configuration file that defines the relevant parameters for this virtual cluster will be written to the file `saved_configuration.xml`. Any Hadoop program that uses the `GenericOptionsParser` may be passed a `-conf saved_configuration.xml` argument, which will cause the program to load the configuration parameters in `saved_configuration.xml`, and to use the virtual cluster for MapReduce and HDFS services.

I find this particularly handy for debugging jobs when I am on the road because the HDFS data persists after the debugger has exited, and I can examine the job status via the web GUIs. The only problem I have is that the per-task log files are not available via the web GUI, and the HDFS files are not available via the web GUI because of issues inside the Hadoop-supplied `MiniMRCluster` code. The following command lists the files in the virtual HDFS:

```
bin/hadoop dfs -conf saved_configuration.xml -ls
```

This came into being when I was trying to work on the unit tests while on the road, using a machine with Windows XP as the host operating system. The virtual clusters would periodically not start, and I became very frustrated. I wrote this and after it started, it stayed running, and I could use it reliably for multiple tests. The ability to examine the data files in HDFS and to interact with the web interfaces was a pleasant discovery.

Eclipse Project for the Example Code

The example code was developed in Eclipse 3.4, and the project and class path files are part of the download, enabling you to load up, experiment with, and run the example code.

Summary

Hadoop is powerful tool for large-scale data processing. Many people and organizations are leveraging the power of Hadoop MapReduce and providing domain-specific package tools. Distributed column-oriented databases are the current mantra of the scalable web services community; and HBase and Hypertable provide them. Data mining, extracting, transforming, and loading without having to write custom MapReduce jobs are provided with Hive and Pig. Machine learning and recognition are provided by Mahout and Hama, and distributed search is provided by the Katta project.

I am partial to the Cloudera Hadoop distribution because it has good support, back ported fixes, training, is free, and is responsive to community needs. Try the various packages discussed in this chapter explore and enjoy.