HDFS

Dr. Yashar Moshfeghi

Big Data 4

Tutorial, Week 2

*Part of the slides are taken from Dr. Nikos Ntarmos

Outline

- Command-line interface
  - hadoop fs/hdfs dfs
- Programmatic access
  - Common parts
  - Basic operations

Command-line interface::hadoop fs/hdfs dfs

Introduction

- There are many other interfaces to HDFS, e.g Web UI
- The command line is one of the simplest and, to many developers, the most familiar.

HDFS CLI: Introduction

- CLI *entry points*:

$ hadoop fs [cmd . . .] → **old and deprecated**
$ hdfs dfs [cmd ...] →**current standard**

- These are powerful commands; use -help and -usage when in doubt!

$ hdfs dfs -help [cmd ...]

*Show **long** help message for given command.*

$ hdfs dfs -usage [cmd ...]

*Show just a **short message** showing usage syntax for given command.*

- Ditch the dash (-) for individual commands; e.g., for command -mkdir:
  - $ hdfs dfs -help -mkdir ×
  - $ hdfs dfs -help mkdir ✓

Example

$ hadoop fs
Usage: java FsShell
[-ls <path>]
[-lsr <path>]
[-df [<path>]]
[-du <path>]
[-dus <path>] [-count[-q] <path>]
[-mv <src> <dst>]
[-cp <src> <dst>]
[-rm [-skipTrash] <path>]
[-rmr [-skipTrash] <path>]
[-expunge]
[-put <localsrc> ... <dst>]
[-copyFromLocal <localsrc> ... <dst>]
[-moveFromLocal <localsrc> ... <dst>]
[-get [-ignoreCrc] [-crc] <src> <localdst>]
[-getmerge <src> <localdst> [addnl]]
[-cat <src>]
[-text <src>]
[-copyToLocal [-ignoreCrc] [-crc] <src> <localdst>]
[-moveToLocal [-crc] <src> <localdst>]
[-mkdir <path>]
[-setrep [-R] [-w] <rep> <path/file>]
[-touchz <path>]
[-test -[ezd] <path>]
[-stat [format] <path>]
[-tail [-f] <file>]
[-chmod [-R] <MODE[,MODE]... | OCTALMODE> PATH...]
[-chown [-R] [OWNER][:[GROUP]] PATH...]
[-chgrp [-R] GROUP PATH...]
[-help [cmd]]

HDFS CLI: File Uploading & Downloading

- Upload file(s):

$ hdfs dfs -put/-copyFromLocal <file1> [<file2> ...] <dst>

$ hdfs dfs -moveFromLocal <file1> [<file2> ...] <dst>

- Download file(s):

$ hdfs dfs -get/-copyToLocal [-ignoreCrc|-crc] <file1> ... <dst>

$ hdfs dfs -moveToLocal ... (*not implemented yet*)

- Download and merge file(s):

$ hdfs dfs -getmerge [-nl] <src dir pattern> <dst>

Example

$ hdfs dfs -copyFromLocal input/docs/quangle.txt hdfs://localhost/user/yashar/quangle.txt

// OR //

$ hdfs dfs -copyFromLocal input/docs/quangle.txt /user/yashar/quangle.txt

URI is specified in *core-site.xml*

// OR //

$ hdfs dfs -copyFromLocal input/docs/quangle.txt quangle.txt

Copy to home directory

$ hdfs dfs -copyToLocal quangle.txt quangle.copy.txt
$ md5 input/docs/quangle.txt quangle.copy.txt

Copy back to local file System

MD5 (input/docs/quangle.txt) = a16f231da6b05e2ba7a339320e7dacd9
MD5 (quangle.copy.txt) = a16f231da6b05e2ba7a339320e7dacd9

HDFS CLI: Directory Creation & List

- Create directory:

$ hdfs dfs -mkdir [-p] <path>

- List files/directories matching pattern:

$ hdfs dfs -ls [-d] [-h] [-R] [<path> ...]

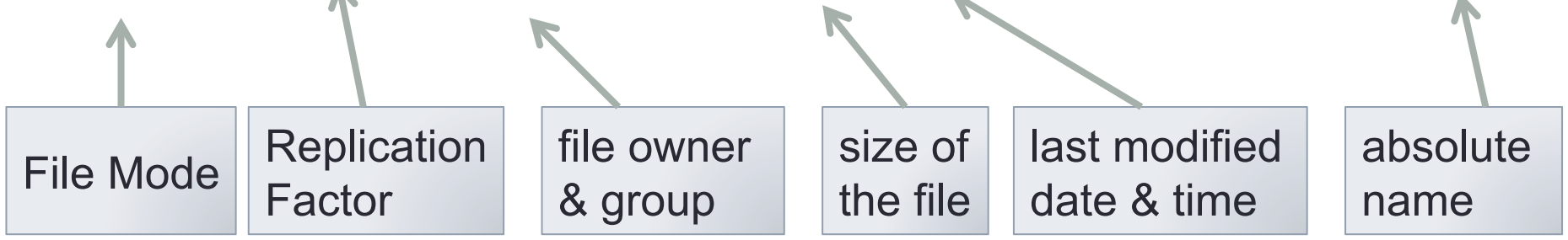
Example

$ hadoop fs -mkdir books

$ hadoop fs -ls .

Directories are treated as metadata and stored by the namenode, not the datanodes

```
Found 2 items
drwxr-xr-x   -  yashar   supergroup   0     2009-04-02 22:41   /user/yashar/books
-rw-r--r--   1  yashar   supergroup   118   2009-04-02 22:29   /user/yashar/quangle.txt
```



traditional Unix filesystem does not have !!

HDFS CLI: Copying, Moving

- Copy/move around a single HDFS namespace:

$ hdfs dfs -cp <file1> [<file2> ...] <dst>

$ hdfs dfs -mv <file1> [<file2> ...] <dst>

These commands allows multiple sources as well in which case the destination must be a directory.

$ hdfs dfs -cp /user/yashar/file1 /user/yashar/file2

$ hdfs dfs -cp /user/yashar/file1 /user/yashar/file2 /user/yashar/dir

HDFS CLI: Deleting

- Delete HDFS file/directory:

$ hdfs dfs -rm [-f] [-r|-R] [-skipTrash] <pattern>

Only deletes non empty directory and files

- Delete *empty* HDFS directory:

$ hdfs dfs -rmdir [-ignore-fail-on-non-empty] <dir>

- Empty HDFS “recycle bin”:

$ hdfs dfs -expunge

Remember !!

Programmatic access :: Common parts

Filesystem reference

First things first: Get a reference to the underlying filesystem.

abstract class org.apache.hadoop.fs.FileSystem

- → org.apache.hadoop.fs.LocalFileSystem
- → **org.apache.hadoop.fs.DistributedFileSystem**
- → org.apache.hadoop.fs.HftpFileSystem
- → org.apache.hadoop.fs.FTPFileSystem
- → org.apache.hadoop.fs.s3.S3FileSystem
- → org.apache.hadoop.fs.kfs.KosmosFileSystem

Note: In general you should strive to write your code against the FileSystem abstract class, to retain portability across filesystems !!

Filesystem reference

```
import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.FileSystem;

Configuration conf = new Configuration();
conf.set("fs.defaultFS", "hdfs://namenode.fqdn:8020");  // Not always necessary
FileSystem fs = FileSystem.get(conf);

[...]

fs.close();
```

- Configuration just a wrapper for java.util.Properties.
- Configuration's default constructor loads information from local conf files.
- Use an explicit namenode URI if executing code on a remote cluster (line 5).
- Use conf.addResource(...) to load non-standard conf (XML) files.

Paths

- Second step: get a reference to a file/directory path
  - org.apache.hadoop.fs.Path

- URI-aware syntax
  - Path f = new Path("file:///user/yashar/file.txt");
  - Path f = new Path("hdfs://localhost:8020/user/yashar/file.txt");
  - Path f = new Path("/user/yashar/file.txt");
  - Path f = new Path("file.txt");

Programmatic access :: Basic operations

Directory/File Creation

```
import org.apache.hadoop.fs.FSDataOutputStream;
[...]

Path dir = new Path("/path/to/dir");
if (fs.mkdirs(dir) == false) // Create directory structure, if not there
            // Error creating directory structure. Bail out...

Path file = new Path("/path/to/dir/file.txt");
// ... or ...
Path file = new Path(dir, "file.txt");

if (fs.exists(file))
// File already exists. Do something...

boolean isCreated = fs.createNewFile(file);
// ... or ...
FSDataOutputStream out = fs.create(file);
// ... or ...
FSDataOutputStream out = fs.create(file, false);
// ... or ...
FSDataOutputStream out = fs.create(file, new FsPermission("u=rw,g=r,o-rwx"),
            true, 1048576, (short)2, 134217728L, null); // ?!?
```

Absolute filename

Filename relative to dir !!

Create empty file

Create/overwrite file

Do not overwrite file.txt

Input

- Read data from a file:

```
import java.io.InputStream;
import org.apache.hadoop.fs.FSDataInputStream;
[...]

InputStream in = fs.open(file);
// ... or ...
FSDataInputStream in = fs.open(file);
String str = in.readUTF();
int i = in.readInt();
double d = in.readDouble();

in.close();
```

- Also check out org.apache.commons.io.IOUtils.

Output

- Write data to a file:

```
import org.apache.hadoop.fs.FSDataOutputStream;
[...]

FSDataOutputStream out = fs.create(file);
out.writeUTF("1024");
out.writeInt(1024);
out.writeDouble(1.0);



out.close();
```

Example

```
public class FileSystemDoubleCat {

        public static void main(String[] args) throws Exception {
                String uri = args[0];
                Configuration conf = new Configuration();
                FileSystem fs = FileSystem.get(URI.create(uri), conf);
                FSDataInputStream in = null;
                try {
                        in = fs.open(new Path(uri));
                        IOUtils.copyBytes(in, System.out, 4096, false);

                }
                finally { IOUtils.closeStream(in);
                }
        }
}
```

Here's the result of running it on a small file:

% **hadoop FileSystemDoubleCat hdfs://localhost/user/yashar/quangle.txt**

On the top of the Crumpetty Tree
The Quangle Wangle sat,
But his face you could not see,

Seeking

- Moving around within a file only supported for reading

```
FSDataInputStream in = fs.open(file);
in.seek(1024); // Go to position 1024 (in bytes)
String someString = in.readUTF(); // Read a string
in.skip(1024); // Skip next 1024 bytes
int position = in.getPos(); // Get current position
in.seek(0); // Go back to start
```

- HDFS built for streaming access ⇒ Seeking not a good idea **(expensive)**!
- Seeking not available for writes ⇒ Only append is supported...

```
FSDataOutputStream out = fs.append(file);
out.writeUTF("some text");
```

... but not by all FileSystem subclasses!

Example

```
public class FileSystemDoubleCat {

        public static void main(String[] args) throws Exception {
                String uri = args[0];
                Configuration conf = new Configuration();
                FileSystem fs = FileSystem.get(URI.create(uri), conf);
                FSDataInputStream in = null;
                try {
                        in = fs.open(new Path(uri));
                        IOUtils.copyBytes(in, System.out, 4096, false);
                        in.seek(0); // go back to the start of the file
                        IOUtils.copyBytes(in, System.out, 4096, false);
                }
                finally { IOUtils.closeStream(in);
                }
        }
}
```

Here's the result of running it on a small file:

% **hadoop FileSystemDoubleCat hdfs://localhost/user/yashar/quangle.txt**

On the top of the Crumpetty Tree
The Quangle Wangle sat,
But his face you could not see,
On account of his Beaver Hat.
On the top of the Crumpetty Tree The Quangle Wangle sat,
But his face you could not see, On account of his Beaver Hat.

Directory/file renaming and deletion

```
Path dir = new Path("/path/to/dir");
Path file = new Path(dir, "file.txt");

// Move file.txt to .../other/dir and rename it
Path dst = new Path("/path/to/other/dir/otherfile.txt");
fs.rename(file, dst);

// Delete the new file
fs.delete(dst);

// Delete the original directory recursively
fs.delete(dir, true);
```

Further Reading

- Apache Hadoop Documentation
  - User Guide:
    http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsUserGuide.html
  - Architecture:
    http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html
  - Command-Line Interface:
    http://hadoop.apache.org/docs/stable/hadoop-project-dist/hadoop-hdfs/HDFSCommands.html
  - Java API:
    http://hadoop.apache.org/docs/stable/api/index.html
    (look under org.apache.hadoop.fs)
- Cloudera Library
  - http://www.cloudera.com/content/cloudera/en/documentation.html#ClouderaDocumentation
- Google . . .