CS 696 Intro to Big Data: Tools and Methods
Fall Semester, 2016
Doc 21 Debugging, WorkFlow, Input
Nov 10, 2016

Assignment for Tuesday Nov 15

Read & understand the examples

- WordMean.java
- WordMedian.java
- WordStandardDeviation.java
- AggregateWordCount.java
- MultiFileWordCount.java

Located in hadoop install

- HADOOP_HOME/share/hadoop/mapreduce/sources/hadoop-mapreduce-examples-2.73-sources

On-line version

https://svn.apache.org/repos/asf/hadoop/common/trunk/hadoop-mapreduce-project/hadoop-mapreduce-examples/src/main/java/org/apache/hadoop/examples/

https://goo.gl/CGdSBz

Examples for Thursday

Sort.java
Grep.java
SecondarySort.java
Join.java

Debugging on a Cluster

System.out
System.err
Logging
Counter
Writing to file

Example

```
private static final Log LOG = LogFactory.getLog(TokenizerMapper.class);
private final static IntWritable one = new IntWritable(1);
private Text word = new Text();

public void map(Object key, Text value, Context context
) throws IOException, InterruptedException {
    StringTokenizer itr = new StringTokenizer(value.toString());
    System.err.println("Error test valye" + value);
    LOG.info("Map key:" + key);
    while (itr.hasMoreTokens()) {
        word.set(itr.nextToken());
        context.write(word, one);
        context.getCounter(WordCounter.WORD).increment(1);
    }
  }
}
```

Syslog

Shuffle Errors

BAD_ID=0

CONNECTION=0

IO_ERROR=0

WRONG_LENGTH=0

WRONG_MAP=0

WRONG_REDUCE=0

WordCount$TokenizerMapper$WordCounter

WORD=8

File Input Format Counters

Bytes Read=50

File Output Format Counters

Bytes Written=41

Hadoop Logs

System daemon logs

HDFS audit logs

MapReduce job history logs

MapReduce task logs - syslog
- Contains file for
  - standard out
  - standard err
- Location given in YARN_LOG_DIR

log aggregation

yarn.log-aggregation-enable

If on YARN moves logs to HDFS

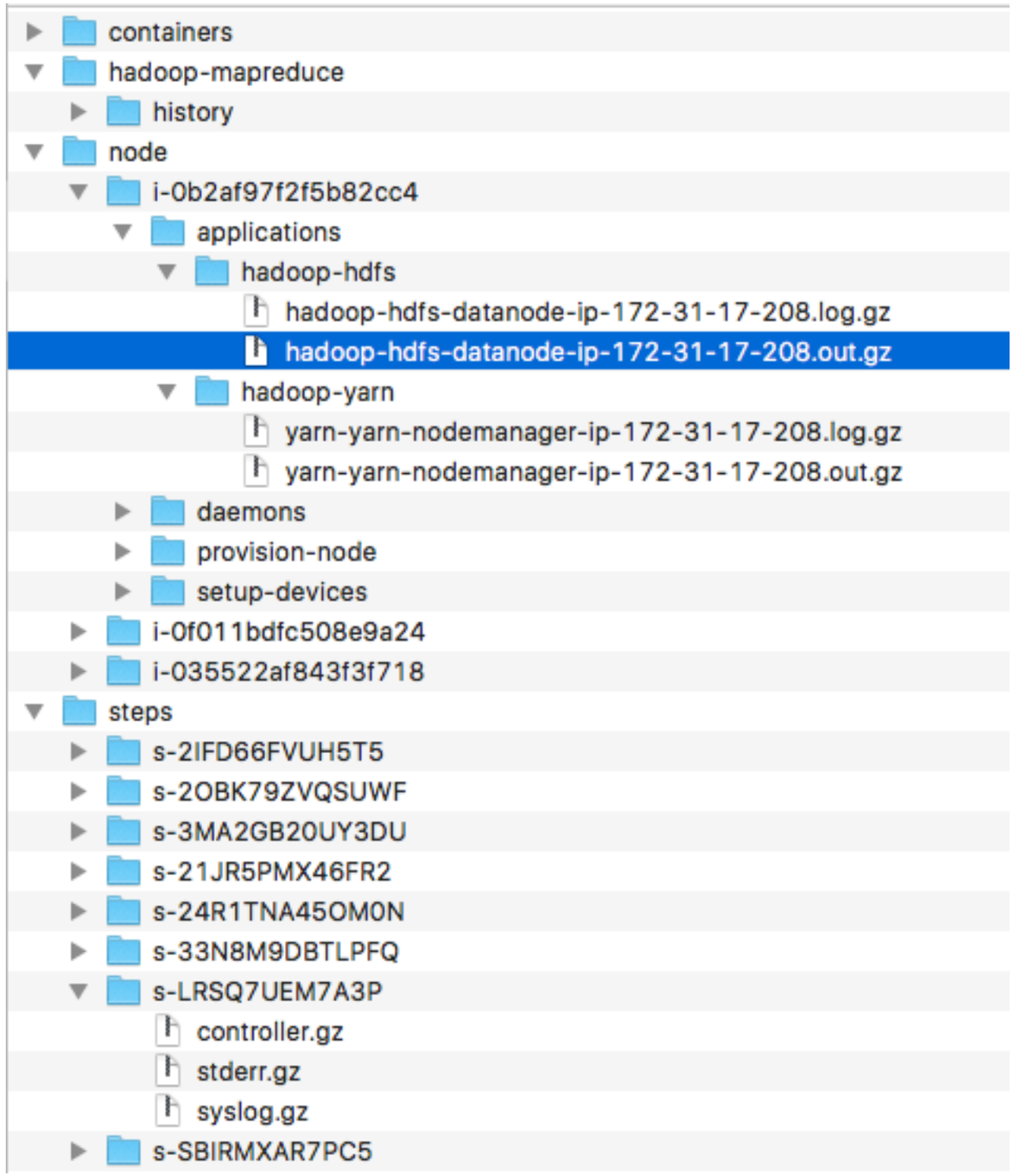
Access them via web UI
mapred job - logs

Off by default

Can access via
http://node-manager-host:8042/logs/userlogs

One can access the local web UI on AWS EMR but it takes some set up

On AWS EMR


containers
hadoop-mapreduce
history
node
i-0b2af97f2f5b82cc4
applications
hadoop-hdfs
hadoop-hdfs-datanode-ip-172-31-17-208.log.gz
hadoop-hdfs-datanode-ip-172-31-17-208.out.gz
hadoop-yarn
yarn-yarn-nodemanager-ip-172-31-17-208.log.gz
yarn-yarn-nodemanager-ip-172-31-17-208.out.gz
daemons
provision-node
setup-devices
i-0f011bdfc508e9a24
i-035522af843f3f718
steps
s-2IFD66FVUH5T5
s-2OBK79ZVQSUWF
s-3MA2GB20UY3DU
s-21JR5PMX46FR2
s-24R1TNA45OM0N
s-33N8M9DBTLPFQ
s-LRSQ7UEM7A3P
controller.gz
stderr.gz
syslog.gz
s-SBIRMXAR7PC5

Writing files

Local file system

- On cluster
  - Where are the files?
  - Where can you write files

HDFS

- Hadoop uses own IO classes not standard Java IO classes

FileSystem

org.apache.hadoop.fs.FileSystem

Main Java class to interact with HDFS

static FileSystem get(URI uri, Configuration conf)

FSDataOutputStream create(Path f)

- There are 12 versions of create
- Creates a file & all need directories in the path

Example - copying local file to HDFS

```
public class FileCopyWithProgress {
  public static void main(String[] args) throws Exception {
    String localSrc = args[0];
    String dst = args[1];

    InputStream in = new BufferedInputStream(new FileInputStream(localSrc));

    Configuration conf = new Configuration();
    FileSystem fs = FileSystem.get(URI.create(dst), conf);
    OutputStream out = fs.create(new Path(dst), new Progressable() {
      public void progress() {
        System.out.print(".");
      }
    });

    IOUtils.copyBytes(in, out, 4096, true);
  }
}
```

FSDataOutputStream

org.apache.hadoop.fs.FSDataOutputStream

Standard Java Stream

write, writeChar, writeInt, writeUTF, etc.

hflush()

hsync()

IOUtils

org.apache.hadoop.io.IOUtils

static void copyBytes(InputStream in, OutputStream out, int buffSize)
5 versions

static void readFully(InputStream in, byte[] buf, int off, int len)

static void writeFully(WritableByteChannel bc, ByteBuffer buf)

FileUtil

org.apache.hadoop.fs.FileUtil

static boolean copy(FileSystem srcFS, Path src, FileSystem dstFS, Path dst, boolean deleteSource, boolean overwrite, Configuration conf)

Copy files between FileSystems.

static boolean copyMerge(FileSystem srcFS, Path srcDir, FileSystem dstFS, Path dstFile, boolean deleteSource, Configuration conf, String addString)

Copy all files in a directory to one output file (merge).

static File createLocalTempFile(File basefile, String prefix, boolean isDeleteOnExit)

Chaining Jobs

Linear sequencing

AWS EMR Steps

Oozie

Linear Sequencing of Jobs

Using ToolRunner

- Create a Job
- Configure the job
- job.waitForCompletion(true)

Create n jobs & run them in sequence

```
Configuration conf = new Configuration();
Job job = Job.getInstance(conf, "word count");
job.setJarByClass(WordCount.class);
job.setMapperClass(TokenizerMapper.class);
job.setCombinerClass(IntSumReducer.class);
job.setReducerClass(IntSumReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(IntWritable.class);
FileInputFormat.addInputPath(job, new Path(args[0]));
FileOutputFormat.setOutputPath(job, new Path(args[1]
job.setOutputValueClass(IntWritable.class);
return job.waitForCompletion(true) ? 0 : 1;
```

```
@Override
public int run(String[] args) throws Exception {

    Configuration conf = new Configuration();
    Job jobA = configureFirstJob(args);
    Boolean wasSuccess = job.waitForCompletion(true) ;
    if (!wasSuccess) { return -1; }
   Job jobB = configureSecondJob(args);
   returnn jobB.waitForCompletion(true) ? 0 : 1
}

public static void main(String[] args) throws Exception {
    int exitCode = ToolRunner.run(new WordCount(), args);
    System.exit(exitCode);
}
```

Job

org.apache.hadoop.mapreduce.Job

Counters getCounters()

JobStatus getStatus()

TaskCompletionEvent[] getTaskCompletionEvents(int startFrom)

boolean isComplete()

void submit()

void killJob()

Example

Look at Grep.java in hadoop examples

AWS EMR Steps

Add step | Clone step

Steps View all interactive

Filter: All steps | Filter steps ... | 10 steps (all loaded)

	ID	Name	Status	Start time (UTC-8)	Elapsed time	Log files
▶	s-295AHS7BUK1GE	WordCount3	Completed	2016-11-09 20:35 (UTC-8)	1 minute	View logs
▶ ●	s-235FT6EBZTQCS	WordCount3	Failed	2016-11-09 20:19 (UTC-8)	16 seconds	controller
▶	s-SBIRMXAR7PC5	WordCount3	Completed	2016-11-09 19:50 (UTC-8)	1 minute	View logs
▶ ●	s-2OBK79ZVQSUWF	WordCount3	Failed	2016-11-09 19:50 (UTC-8)	20 seconds	controller
▶ ●	s-33N8M9DBTLPFQ	WordCount3	Failed	2016-11-09 19:50 (UTC-8)	14 seconds	controller
▶ ●	s-2IFD66FVUH5T5	WordCount4	Failed	2016-11-09 19:49 (UTC-8)	14 seconds	controller
▶	s-21JR5PMX46FR2	WordCount3	Completed	2016-11-09 19:48 (UTC-8)	1 minute	View logs
▶ ●	s-LRSQ7UEM7A3P	WordCount2	Failed	2016-11-09 19:47 (UTC-8)	26 seconds	controller
▶ ●	s-24R1TNA45OM0N	WordCount2	Failed	2016-11-09 19:47 (UTC-8)	4 seconds	controller
▶	s-3MA2GB20UY3DU	Setup hadoop debugging	Completed	2016-11-09 19:47 (UTC-8)	2 seconds	View logs

AWS EMR Steps

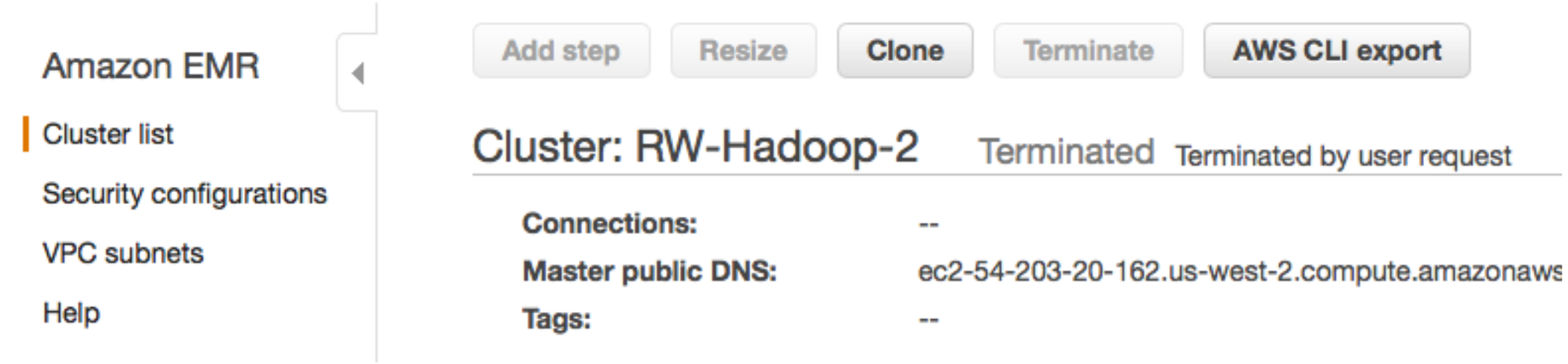
Steps are done in order

Using AWS EMR Web interface

- Create cluster
- Add steps one at time
- Can have step terminate on failure

Can clone cluster setting to rerun set of programs

AWS CLI interface



Example

```
aws emr create-cluster --applications Name=Hadoop --ec2-attributes
    '{"KeyName":"hadoop","InstanceProfile":"EMR_EC2_DefaultRole",
    "SubnetId":"subnet-0f55196b","EmrManagedSlaveSecurityGroup":"sg-65bffa1c",
    "EmrManagedMasterSecurityGroup":"sg-62bffa1b"}'

    --service-role EMR_DefaultRole --enable-debugging --release-label emr-5.1.0
    --log-uri 's3n://rw-hadoop-logs/elasticmapreduce/'
    --steps '[{"Args":["s3://rw-wc-input-data/*","s3://rw-wc-output-data/try9"],
        "Type":"CUSTOM_JAR","ActionOnFailure":"CONTINUE",
        "Jar":"s3://rw-hadoop-jars/ConfigureExample.jar","Properties":"","Name":"Wo
Count"}]'
    --name 'RW-Hadoop-2' --instance-groups '[{"InstanceCount":1,
        "InstanceGroupType":"MASTER","InstanceType":"m1.medium",
        "Name":"Master instance group - 1"},
        {"InstanceCount":2,"InstanceGroupType":"CORE",
         "InstanceType":"m1.medium","Name":"Core instance group - 2"}]' --region us-west-2
```

AWS CLI

https://aws.amazon.com/cli/

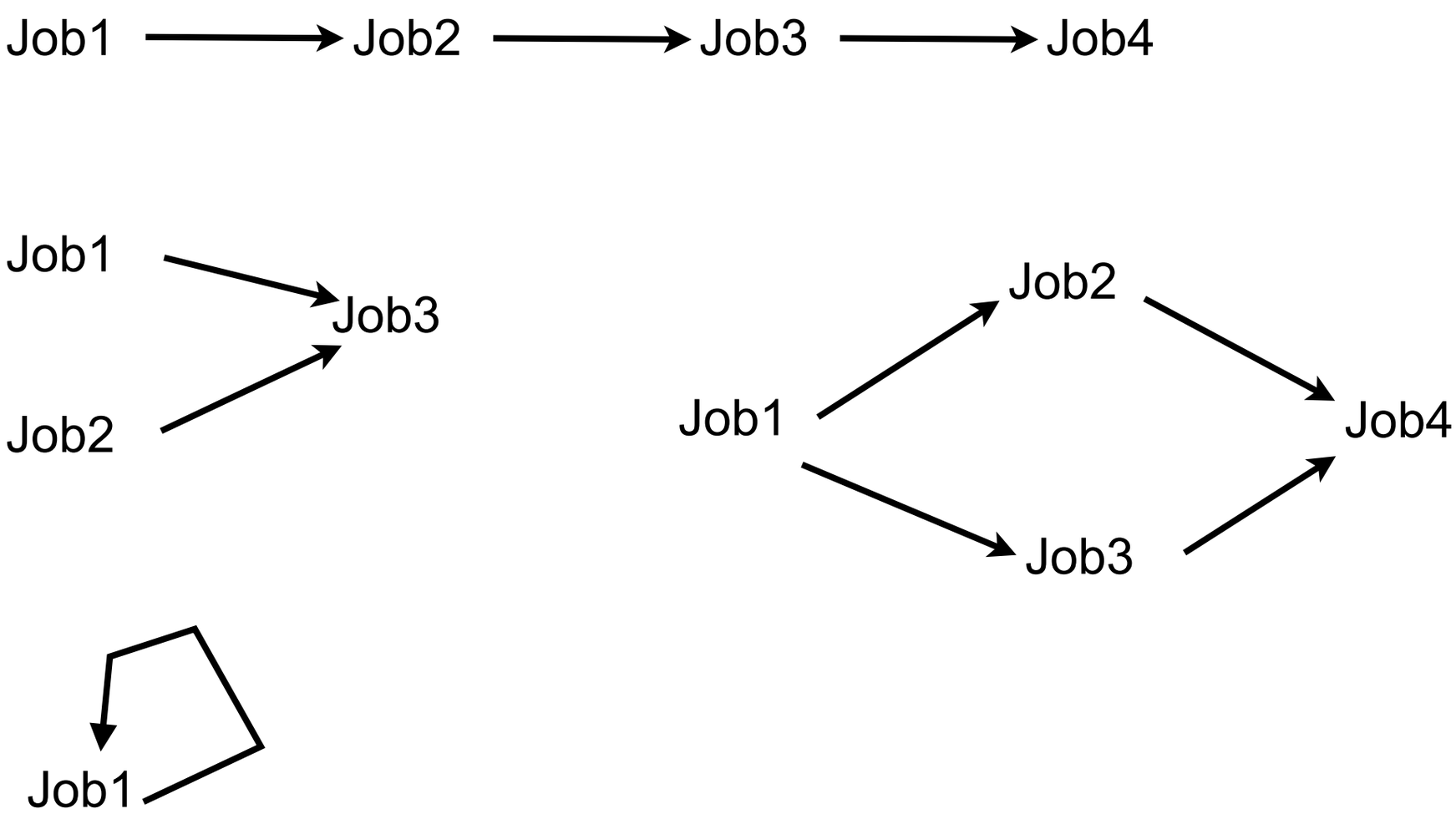
Command Line interface

Installed on Amazon's linux cluster machines
    ssh into cluster after getting public/private keys

Local client
    Windows, Mac, Linux

aws ec2 start-instances --instance-ids i-1348636c

Job Pipelines



Oozie

System for running workflows of dependent jobs

AWS EMR

- Supports Oozie, but need to use AWS CLI

Works with

- MapReduce, Pig, Hive, Spark, etc

Action Nodes

- email, shell, hive, Sqoop, mapreduce, Spark, hdfs, java program, streaming

Control-flow Nodes

- Uses Hadoop Process Defintion Language

Oozie

Uses XML to define

- Action Nodes
- Flow-Control nodes

```
<map-reduce>
 <job-tracker>${resourceManager}</job-tracker>
 <name-node>${nameNode}</name-node>
 <prepare>
  <delete path="${nameNode}/user/${wf:user()}/output"/>
 </prepare>
 <configuration>
  <property>
   <name>mapred.mapper.new-api</name>
   <value>true</value>
  </property>
  <property>
   <name>mapred.reducer.new-api</name>
   <value>true</value>
  </property>
  <property>
   <name>mapreduce.job.map.class</name>
   <value>MaxTemperatureMapper</value>
  </property>
  <property>
   <name>mapreduce.job.combine.class</name>
   <value>MaxTemperatureReducer</value>
  </property>
  <property>
   <name>mapreduce.job.reduce.class</name>
   <value>MaxTemperatureReducer</value>
  </property>
  <property>
   <name>mapreduce.job.output.key.class</name>
   <value>org.apache.hadoop.io.Text</value>
  </property>
  <property>
   <name>mapreduce.job.output.value.class</name>
   <value>org.apache.hadoop.io.IntWritable</value>
  </property>
  <property>
   <name>mapreduce.input.fileinputformat.inputdir</name>
   <value>/user/${wf:user()}/input/ncdc/micro</value>
  </property>
  <property>
   <name>mapreduce.output.fileoutputformat.outputdir</name>
   <value>/user/${wf:user()}/output</value>
  </property>
 </configuration>
</map-reduce>
<ok to="end"/>
<error to="fail"/>
</action>
<kill name="fail">
 <message>MapReduce failed, error message[${wf:errorMessage(wf:lastErrorNode())}]
 </message>
</kill>
<end name="end"/>
</workflow-app>
```

Map

Input to map so far has been

Key - offset in file

Value - line of text

What if we want output of on hadoop job to be input of second?

Bye	1
Goodbye	1

Key to be word

Value to be Integer value

Input Formats

Hadoop supports multiple data formats

Can implement classes to support new format

split
    Input processed by one map

record
    split is divided into records
    each record is key-value pair

```
public abstract class InputSplit {
  public abstract long getLength() throws IOException, InterruptedException;
  public abstract String[] getLocations() throws IOException,
     InterruptedException;
}
```

InputFormat

```
public abstract class InputFormat<K, V> {
  public abstract List<InputSplit> getSplits(JobContext context)
      throws IOException, InterruptedException;

  public abstract RecordReader<K, V>
      createRecordReader(InputSplit split, TaskAttemptContext context)
        throws IOException, InterruptedException;
}
```

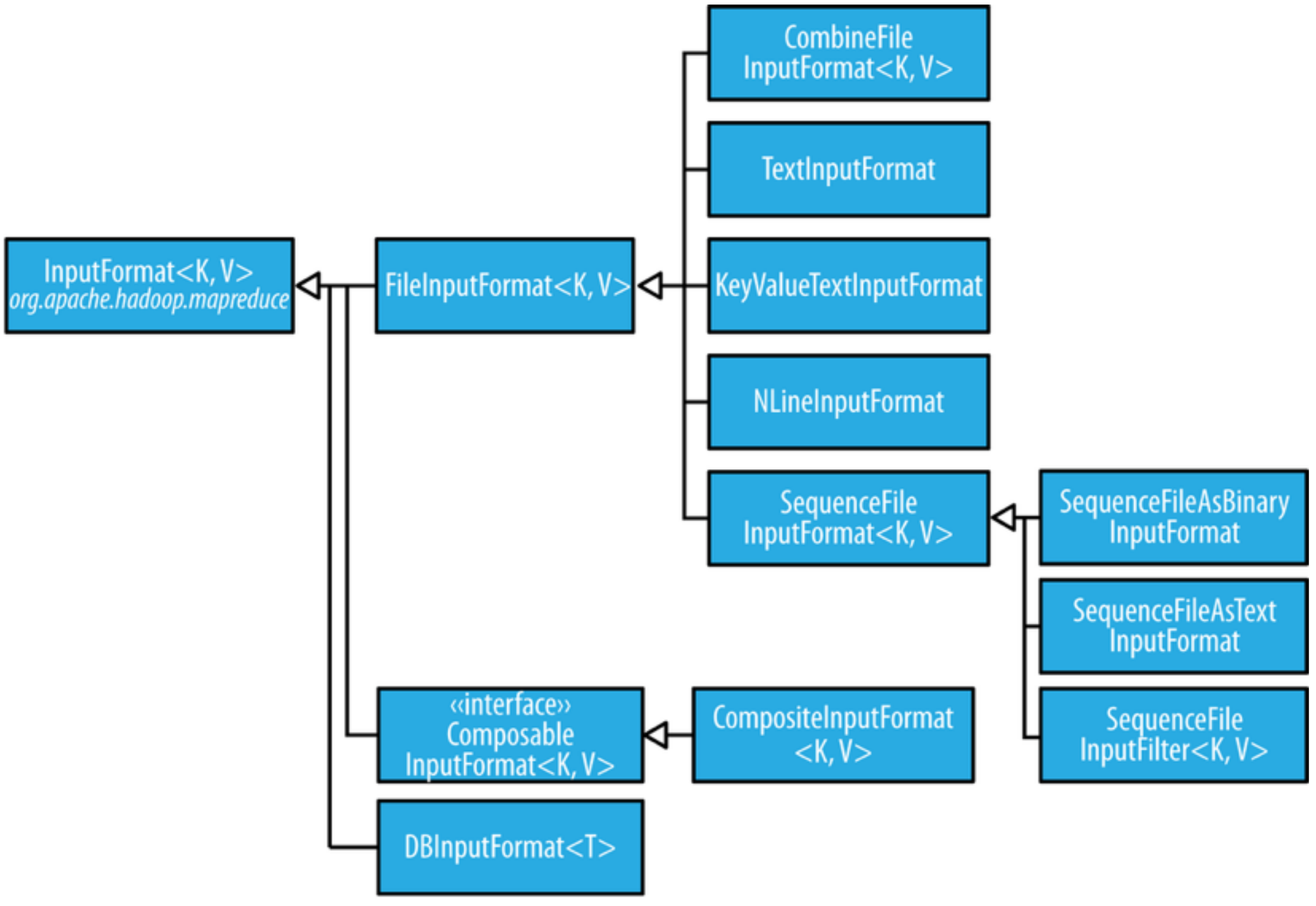
Hadoop

Calls getSplits to divide input into splits

For each split call createRecordReader

Iterate over reader calling your map function each time

```
public void run(Context context) throws IOException, InterruptedException {
  setup(context);
  while (context.nextKeyValue()) {
    map(context.getCurrentKey(), context.getCurrentValue(), context);
  }
  cleanup(context);
}
```


InputFormat<K, V>
org.apache.hadoop.mapreduce
FileInputFormat<K, V>
CombineFile InputFormat<K, V>
TextInputFormat
KeyValueTextInputFormat
NLineInputFormat
SequenceFile InputFormat<K, V>
SequenceFileAsBinary InputFormat
SequenceFileAsText InputFormat
SequenceFile InputFilter<K, V>
«interface» Composable InputFormat<K, V>
CompositeInputFormat <K, V>
DBInputFormat<T>

FileInputFormat

Which files are input to job

- lists of files & directories
- Different ways of specifying
- Filters can select files from lists

Generating splits

- Subclasses define records

Splits

Only large files are split

dfs.blocksize

mapreduce.input.fileinputformat.split.maxsize

mapreduce.input.fileinputformat.split.minsize

split size

max(minSize, min(maxSize, blockSize))

Small Files

Hadoop does not work well with small files

org.apache.hadoop.mapreduce.lib.input.CombineFileInputFormat<K,V>

Abstract class for combining files into larger files

Example

MultiFileWordCount.java in hadoop examples