Cloudera CCD-470

Cloudera Certified Developer for Apache Hadoop CDH4 Upgrade Exam

Version: 3.0

QUESTION NO: 1

When is the earliest point at which the reduce method of a given Reducer can be called?

A. As soon as at least one mapper has finished processing its input split.
B. As soon as a mapper has emitted at least one record.
C. Not until all mappers have finished processing all records.
D. It depends on the InputFormat used for the job.

Answer: C

Explanation: In a MapReduce job reducers do not start executing the reduce method until the all Map jobs have completed. Reducers start copying intermediate key-value pairs from the mappers as soon as they are available. The programmer defined reduce method is called only after all the mappers have finished.

Note: The reduce phase has 3 steps: shuffle, sort, reduce. Shuffle is where the data is collected by the reducer from each mapper. This can happen while mappers are generating data since it is only a data transfer. On the other hand, sort and reduce can only start once all the mappers are done.

Why is starting the reducers early a good thing? Because it spreads out the data transfer from the mappers to the reducers over time, which is a good thing if your network is the bottleneck.

Why is starting the reducers early a bad thing? Because they "hog up" reduce slots while only copying data. Another job that starts later that will actually use the reduce slots now can't use them.

You can customize when the reducers startup by changing the default value of mapred.reduce.slowstart.completed.maps in mapred-site.xml. A value of 1.00 will wait for all the mappers to finish before starting the reducers. A value of 0.0 will start the reducers right away. A value of 0.5 will start the reducers when half of the mappers are complete. You can also change mapred.reduce.slowstart.completed.maps on a job-by-job basis.

Typically, keep mapred.reduce.slowstart.completed.maps above 0.9 if the system ever has multiple jobs running at once. This way the job doesn't hog up reducers when they aren't doing anything but copying data. If you only ever have one job running at a time, doing 0.1 would probably be appropriate.

Reference: 24 Interview Questions & Answers for Hadoop MapReduce developers, When is the reducers are started in a MapReduce job?

QUESTION NO: 2

Which describes how a client reads a file from HDFS?

A. The client queries the NameNode for the block location(s). The NameNode returns the block location(s) to the client. The client reads the data directory off the DataNode(s).
B. The client queries all DataNodes in parallel. The DataNode that contains the requested data responds directly to the client. The client reads the data directly off the DataNode.
C. The client contacts the NameNode for the block location(s). The NameNode then queries the DataNodes for block locations. The DataNodes respond to the NameNode, and the NameNode redirects the client to the DataNode that holds the requested data block(s). The client then reads the data directly off the DataNode.
D. The client contacts the NameNode for the block location(s). The NameNode contacts the DataNode that holds the requested data block. Data is transferred from the DataNode to the NameNode, and then from the NameNode to the client.

Answer: C
Explanation: The Client communication to HDFS happens using Hadoop HDFS API. Client applications talk to the NameNode whenever they wish to locate a file, or when they want to add/copy/move/delete a file on HDFS. The NameNode responds the successful requests by returning a list of relevant DataNode servers where the data lives. Client applications can talk directly to a DataNode, once the NameNode has provided the location of the data.
Reference: 24 Interview Questions & Answers for Hadoop MapReduce developers, How the Client communicates with HDFS?

QUESTION NO: 3

You are developing a combiner that takes as input Text keys, IntWritable values, and emits Text keys, IntWritable values. Which interface should your class implement?

A. Combiner <Text, IntWritable, Text, IntWritable>
B. Mapper <Text, IntWritable, Text, IntWritable>
C. Reducer <Text, Text, IntWritable, IntWritable>
D. Reducer <Text, IntWritable, Text, IntWritable>
E. Combiner <Text, Text, IntWritable, IntWritable>

Answer: D
Explanation:

QUESTION NO: 4

Indentify the utility that allows you to create and run MapReduce jobs with any executable or script as the mapper and/or the reducer?

A. Oozie
B. Sqoop
C. Flume
D. Hadoop Streaming
E. mapred

Answer: D

Explanation: Hadoop streaming is a utility that comes with the Hadoop distribution. The utility allows you to create and run Map/Reduce jobs with any executable or script as the mapper and/or the reducer.
Reference:http://hadoop.apache.org/common/docs/r0.20.1/streaming.html(Hadoop Streaming, second sentence)

QUESTION NO: 5

How are keys and values presented and passed to the reducers during a standard sort and shuffle phase of MapReduce?

A. Keys are presented to reducer in sorted order; values for a given key are not sorted.
B. Keys are presented to reducer in sorted order; values for a given key are sorted in ascending order.
C. Keys are presented to a reducer in random order; values for a given key are not sorted.
D. Keys are presented to a reducer in random order; values for a given key are sorted in ascending order.

Answer: A

Explanation: Reducer has 3 primary phases:

1. Shuffle
The Reducer copies the sorted output from each Mapper using HTTP across the network.

2. Sort
The framework merge sorts Reducer inputs by keys (since different Mappers may have output the same key).

The shuffle and sort phases occur simultaneously i.e. while outputs are being fetched they are merged.

SecondarySort

To achieve a secondary sort on the values returned by the value iterator, the application should extend the key with the secondary key and define a grouping comparator. The keys will be sorted using the entire key, but will be grouped using the grouping comparator to decide which keys and values are sent in the same call to reduce.

3. Reduce

In this phase the reduce(Object, Iterable, Context) method is called for each <key, (collection of values)> in the sorted inputs.

The output of the reduce task is typically written to a RecordWriter via TaskInputOutputContext.write(Object, Object).

The output of the Reducer is not re-sorted.

Reference: org.apache.hadoop.mapreduce, Class
Reducer<KEYIN,VALUEIN,KEYOUT,VALUEOUT>

QUESTION NO: 6

Assuming default settings, which best describes the order of data provided to a reducer’s reduce method:

A. The keys given to a reducer aren’t in a predictable order, but the values associated with those keys always are.
B. Both the keys and values passed to a reducer always appear in sorted order.
C. Neither keys nor values are in any predictable order.
D. The keys given to a reducer are in sorted order but the values associated with each key are in no predictable order

Answer: D

Explanation: Reducer has 3 primary phases:

1. Shuffle

The Reducer copies the sorted output from each Mapper using HTTP across the network.

2. Sort

The framework merge sorts Reducer inputs by keys (since different Mappers may have output the same key).

The shuffle and sort phases occur simultaneously i.e. while outputs are being fetched they are merged.

SecondarySort
To achieve a secondary sort on the values returned by the value iterator, the application should extend the key with the secondary key and define a grouping comparator. The keys will be sorted using the entire key, but will be grouped using the grouping comparator to decide which keys and values are sent in the same call to reduce.

3. Reduce
In this phase the reduce(Object, Iterable, Context) method is called for each <key, (collection of values)> in the sorted inputs.

The output of the reduce task is typically written to a RecordWriter via TaskInputOutputContext.write(Object, Object).

The output of the Reducer is not re-sorted.

Reference: org.apache.hadoop.mapreduce, Class
Reducer<KEYIN,VALUEIN,KEYOUT,VALUEOUT>

QUESTION NO: 7

You wrote a map function that throws a runtime exception when it encounters a control character in input data. The input supplied to your mapper contains twelve such characters totals, spread across five file splits. The first four file splits each have two control characters and the last split has four control characters.

Indentify the number of failed task attempts you can expect when you run the job with mapred.max.map.attempts set to 4:

A. You will have forty-eight failed task attempts
B. You will have seventeen failed task attempts
C. You will have five failed task attempts
D. You will have twelve failed task attempts
E. You will have twenty failed task attempts

Answer: E

Explanation: There will be four failed task attempts for each of the five file splits.

Note:
Whenthejobtracker is notified of a task attempt that has failed (by the tasktracker's heartbeat tall), it will reschedule execution of the task. The jobtracker will try to avoid rescheduling the task on a tasktracker where ithas previously tailed. Furthermore, if a task fails four times (or more), itwillnot be retried further. This value is configurable:themaximum number of attempts to run a task is controlled by the mapred.map.max.attempts property for map tasks and mapred.reduce.max.attempts for reduce tasks. By default, ifany task fails four times (or whatever the maximum number of attempts is configured to), the whole job fails.

QUESTION NO: 8

You want to populate an associative array in order to perform a map-side join. You’ve decided to put this information in a text file, place that file into the DistributedCache and read it in your Mapper before any records are processed.

Indentify which method in the Mapper you should use to implement code for reading the file and populating the associative array?

A. combine
B. map
C. init
D. configure

Answer: D
Explanation: See 3) below.

Here is an illustrative example on how to use the DistributedCache:
// Setting up the cache for the application

1. Copy the requisite files to the FileSystem:

```
$ bin/hadoop fs -copyFromLocal lookup.dat /myapp/lookup.dat
$ bin/hadoop fs -copyFromLocal map.zip /myapp/map.zip
$ bin/hadoop fs -copyFromLocal mylib.jar /myapp/mylib.jar
$ bin/hadoop fs -copyFromLocal mytar.tar /myapp/mytar.tar
$ bin/hadoop fs -copyFromLocal mytgz.tgz /myapp/mytgz.tgz
$ bin/hadoop fs -copyFromLocal mytargz.tar.gz /myapp/mytargz.tar.gz
```

2. Setup the application's JobConf:

```
JobConf job = new JobConf();
DistributedCache.addCacheFile(new URI("/myapp/lookup.dat#lookup.dat"),
job);
DistributedCache.addCacheArchive(new URI("/myapp/map.zip", job);
DistributedCache.addFileToClassPath(new Path("/myapp/mylib.jar"), job);
DistributedCache.addCacheArchive(new URI("/myapp/mytar.tar", job);
DistributedCache.addCacheArchive(new URI("/myapp/mytgz.tgz", job);
DistributedCache.addCacheArchive(new URI("/myapp/mytargz.tar.gz", job);
```

3. Use the cached files in theMapper
orReducer:

```
public static class MapClass extends MapReduceBase
implements Mapper<K, V, K, V> {

private Path[] localArchives;
private Path[] localFiles;

public void configure(JobConf job) {
// Get the cached archives/files
localArchives = DistributedCache.getLocalCacheArchives(job);
localFiles = DistributedCache.getLocalCacheFiles(job);
}

public void map(K key, V value,
OutputCollector<K, V> output, Reporter reporter)
throws IOException {
// Use data from the cached archives/files here
// ...
// ...
output.collect(k, v);
}
}
```

Reference: org.apache.hadoop.filecache , Class DistributedCache

QUESTION NO: 9

You've written a MapReduce job that will process 500 million input records and generated 500 million key-value pairs. The data is not uniformly distributed. Your MapReduce job will create a significant amount of intermediate data that it needs to transfer between mappers and reduces which is a potential bottleneck. A custom implementation of which interface is most likely to reduce the amount of intermediate data transferred across the network?

A. Partitioner
B. OutputFormat
C. WritableComparable
D. Writable
E. InputFormat
F. Combiner

Answer: F

Explanation: Combiners are used to increase the efficiency of a MapReduce program. They are used to aggregate intermediate map output locally on individual mapper outputs. Combiners can help you reduce the amount of data that needs to be transferred across to the reducers. You can use your reducer code as a combiner if the operation performed is commutative and associative.

QUESTION NO: 10

Can you use MapReduce to perform a relational join on two large tables sharing a key? Assume that the two tables are formatted as comma-separated files in HDFS.

A. Yes.
B. Yes, but only if one of the tables fits into memory
C. Yes, so long as both tables fit into memory.
D. No, MapReduce cannot perform relational operations.
E. No, but it can be done with either Pig or Hive.

Answer: A

Explanation: Note:

* Join Algorithms in MapReduce

A) Reduce-side join

B) Map-side join

C) In-memory join

/ Striped Striped variant variant

/ Memcached variant

* Which join to use?
/ In-memory join > map-side join > reduce-side join
/ Limitations of each?
In-memory join: memory
Map-side join: sort order and partitioning
Reduce-side join: general purpose

QUESTION NO: 11

You have just executed a MapReduce job. Where is intermediate data written to after being emitted from the Mapper's map method?

A. Intermediate data in streamed across the network from Mapper to the Reduce and is never written to disk.
B. Into in-memory buffers on the TaskTracker node running the Mapper that spill over and are written into HDFS.
C. Into in-memory buffers that spill over to the local file system of the TaskTracker node running the Mapper.
D. Into in-memory buffers that spill over to the local file system (outside HDFS) of the TaskTracker node running the Reducer
E. Into in-memory buffers on the TaskTracker node running the Reducer that spill over and are written into HDFS.

Answer: D
Explanation: The mapper output (intermediate data) is stored on the Local file system (NOT HDFS) of each individual mapper nodes. This is typically a temporary directory location which can be setup inconfig by the hadoop administrator. The intermediate data is cleaned up after the Hadoop Job completes.

Reference: 24 Interview Questions & Answers for Hadoop MapReduce developers, Where is the Mapper Output (intermediate kay-value data) stored ?

QUESTION NO: 12

You want to understand more about how users browse your public website, such as which pages