MapReduce 原理及操作

该系列课程是应邀实验楼整理编写的，这里需要赞一下实验楼提供了学习的新方式，可以边看博客边上机实验，课程地址为 https://www.shiyanlou.com/courses/237

【注】该系列所使用到安装包、测试数据和代码均可在百度网盘下载，具体地址为 http://pan.baidu.com/s/10PnDs，下载该 PDF 文件

1 环境说明

部署节点操作系统为 CentOS，防火墙和 SElinux 禁用，创建了一个 shiyanlou 用户并在系统根目录下创建/app 目录，用于存放 Hadoop 等组件运行包。因为该目录用于安装 hadoop 等组件程序，用户对 shiyanlou 必须赋予 rwx 权限（一般做法是 root 用户在根目录下创建/app 目录，并修改该目录拥有者为 shiyanlou(chown –R shiyanlou:shiyanlou /app)。

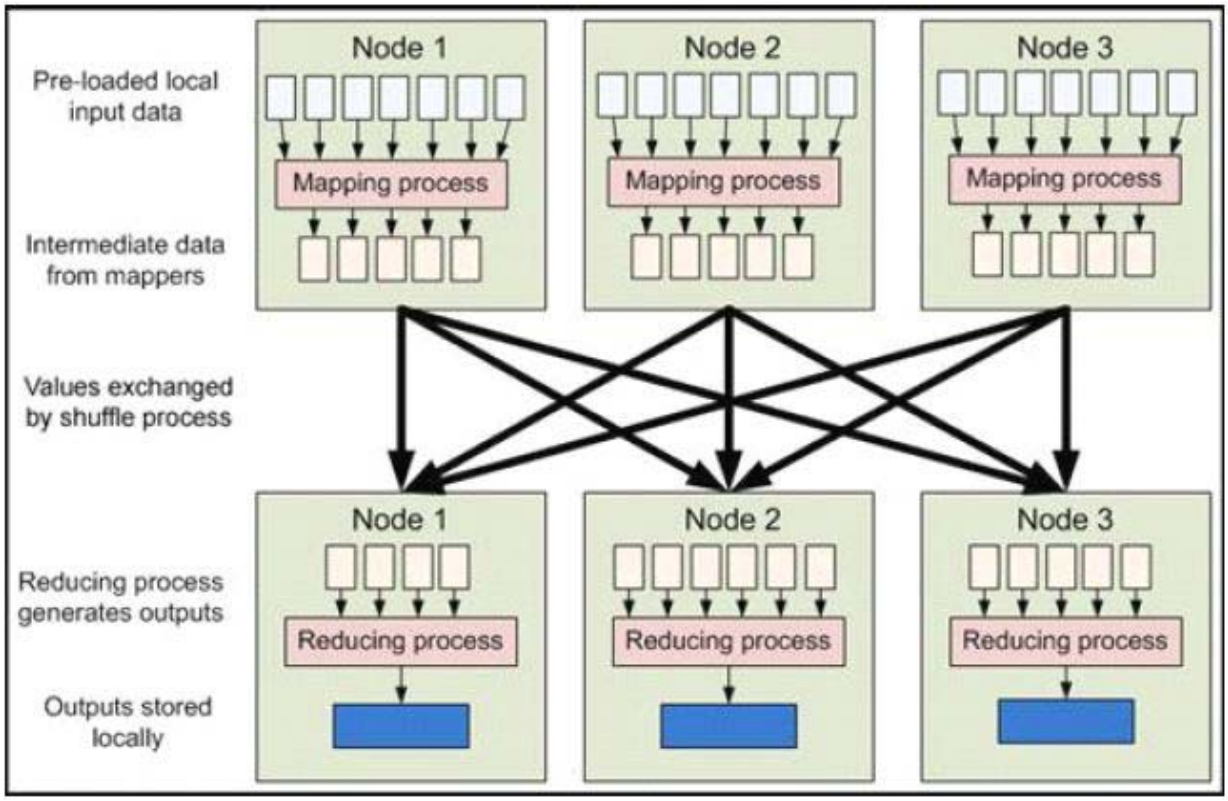
Hadoop 搭建环境：

- 虚拟机操作系统： CentOS6.6 64 位，单核，1G 内存
- JDK：1.7.0_55 64 位
- Hadoop：1.1.2

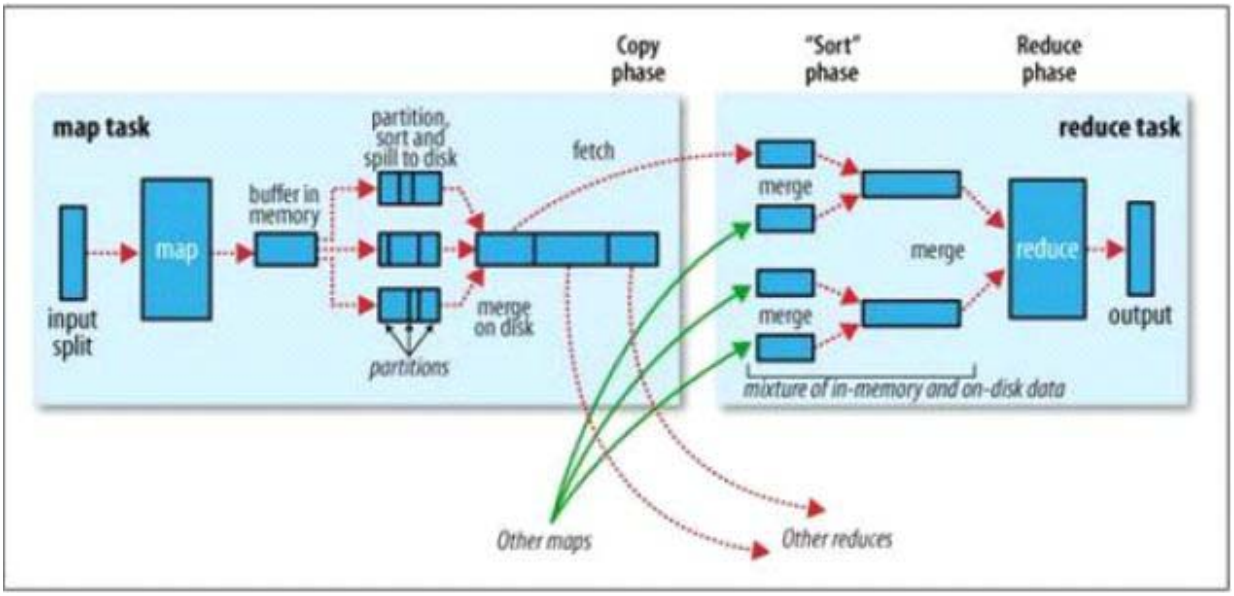
2 MapReduce 原理

2.1 MapReduce 简介

MapReduce 是现今一个非常流行的分布式计算框架，它被设计用于并行计算海量数据。第一个提出该技术框架的是 Google 公司，而 Google 的灵感则来自于函数式编程语言，如 LISP，Scheme，ML 等。MapReduce 框架的核心步骤主要分两部分：Map 和 Reduce。当你向 MapReduce 框架提交一个计算作业时，它会首先把计算作业拆分成若干个 Map 任务，然后分配到不同的节点上去执行，每一个 Map 任务处理输入数据中的一部分，当 Map 任务完成后，它会生成一些中间文件，这些中间文件将会作为 Reduce 任务的输入数据。Reduce 任务的主要目标就是把前面若干个 Map 的输出汇总到一起并输出。从高层抽象来看，MapReduce 的数据流图如下图所示：



2.2 MapReduce 流程分析



2.2.1 Map 过程

1．每个输入分片会让一个 map 任务来处理，默认情况下，以 HDFS 的一个块的大小（默认为 64M）为一个分片，当然我们也可以设置块的大小。map 输出的结果会暂且放在一个环形内存缓冲区中（该缓冲区的大小默认为 100M，由 io.sort.mb 属性控制），当该缓冲区快要溢出时（默认为缓冲区大小的 80%，由 io.sort.spill.percent 属性控制），会在本地文件系统中创建一个溢出文件，将该缓冲区中的数据写入这个文件；

2．在写入磁盘之前，线程首先根据 reduce 任务的数目将数据划分为相同数目的分区，也就是一个 reduce 任务对应一个分区的数据。这样做是为了避免有些 reduce 任务分配到大量数据，而有些 reduce 任务却分到很少数据，甚至没有分到数据的尴尬局面。其实分区就是对数据进行 hash 的过程。然后对每个分区中的数据进行排序，如果此时设置了 Combiner，将排序后的结果进行 Combia 操作，这样做的目的是让尽可能少的数据写入到磁盘；

3．当 map 任务输出最后一个记录时，可能会有很多的溢出文件，这时需要将这些文件合并。合并的过程中会不断地进行排序和 combia 操作，目的有两个：

- 尽量减少每次写入磁盘的数据量
- 尽量减少下一复制阶段网络传输的数据量。最后合并成了一个已分区且已排序的文件。为了减少网络传输的数据量，这里可以将数据压缩，只要将 mapred.compress.map.out 设置为 true 就可以了

4．将分区中的数据拷贝给相对应的 reduce 任务。有人可能会问：分区中的数据怎么知道它对应的 reduce 是哪个呢？其实 map 任务一直和其父 TaskTracker 保持联系，而 TaskTracker 又一直和 JobTracker 保持心跳。所以 JobTracker 中保存了整个集群中的宏观信息。只要 reduce 任务向 JobTracker 获取对应的 map 输出位置就可以了。

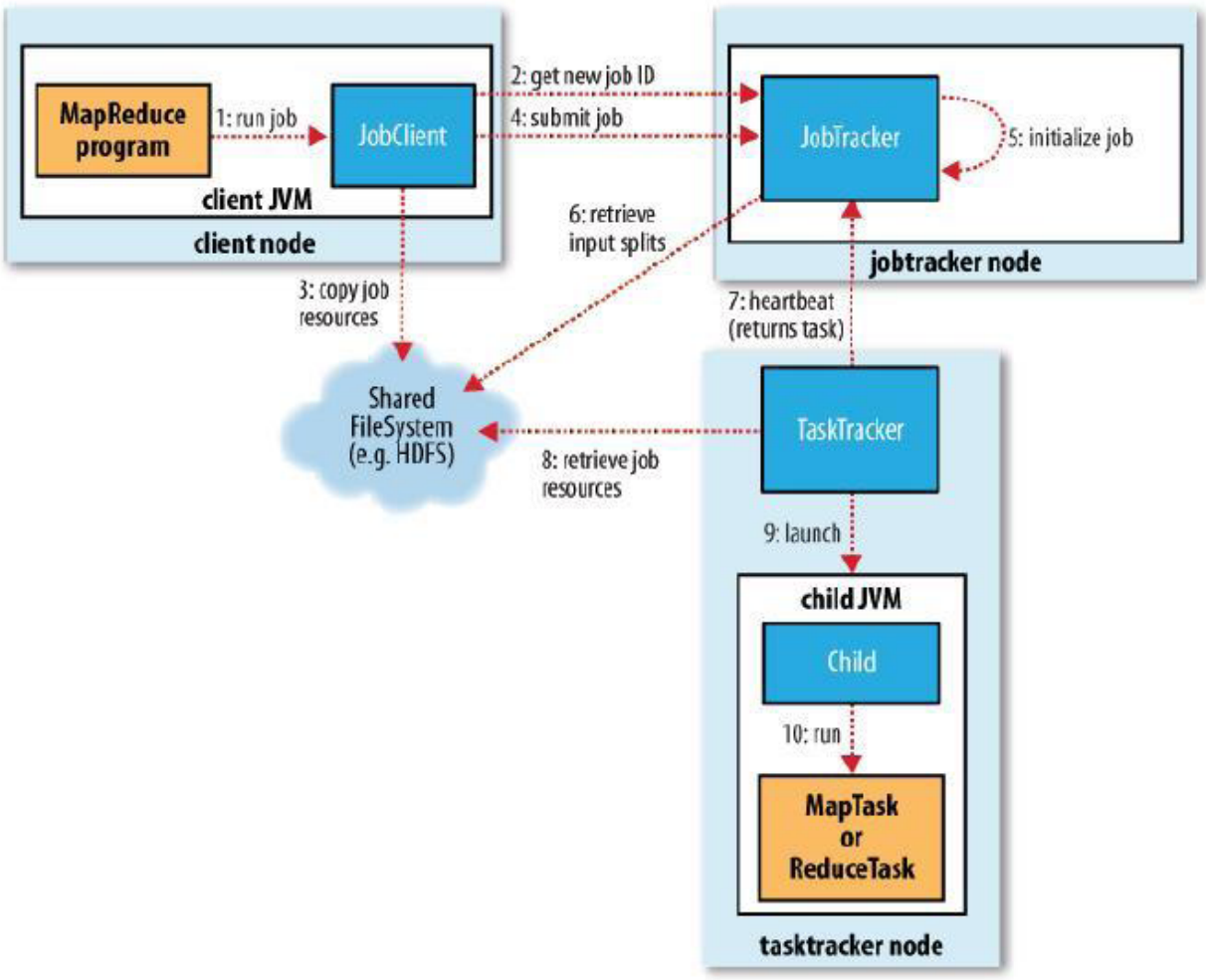
2.2.2 Reduce 过程

1．Reduce 会接收到不同 map 任务传来的数据，并且每个 map 传来的数据都是有序的。如果 reduce 端接受的数据量相当小，则直接存储在内存中（缓冲区大小由 mapred.job.shuffle.input.buffer.percent 属性控制，表示用作此用途的堆空间的百分比），如果数据量超过了该缓冲区大小的一定比例（由 mapred.job.shuffle.merge.percent 决定），则对数据合并后溢写到磁盘中；

2．随着溢写文件的增多，后台线程会将它们合并成一个更大的有序的文件，这样做是为了给后面的合并节省时间。其实不管在 map 端还是 reduce 端，MapReduce 都是反复地执行排序，合并操作；

3．合并的过程中会产生许多的中间文件（写入磁盘了），但 MapReduce 会让写入磁盘的数据尽可能地少，并且最后一次合并的结果并没有写入磁盘，而是直接输入到 reduce 函数。

2.3 MapReduce 工作机制剖析



1. 在集群中的任意一个节点提交 MapReduce 程序；
2. JobClient 收到作业后，JobClient 向 JobTracker 请求获取一个 Job ID；
3. 将运行作业所需要的资源文件复制到 HDFS 上(包括 MapReduce 程序打包的 JAR 文件、配置文件和客户端计算所得的输入划分信息),这些文件都存放在 JobTracker 专门为该作业创建的文件夹中，文件夹名为该作业的 Job ID；
4. 获得作业 ID 后，提交作业；
5. JobTracker 接收到作业后，将其放在一个作业队列里，等待作业调度器对其进行调度，当作业调度器根据自己的调度算法调度到该作业时 ,会根据输入划分信息为每个划分创建一个 map 任务，并将 map 任务分配给 TaskTracker 执行；
6. 对于 map 和 reduce 任务，TaskTracker 根据主机核的数量和内存的大小有固定数量的 map 槽和 reduce 槽。这里需要强调的是：map 任务不是随随便便地分配给某个 TaskTracker 的，这里有个概念叫：数据本地化 (Data-Local)。意思是：将 map 任务分配给含有该 map 处理的数据块的 TaskTracker 上，同时将程序 JAR 包复制到该

TaskTracker 上来运行，这叫“运算移动，数据不移动”；

7. TaskTracker 每隔一段时间会给 JobTracker 发送一个心跳，告诉 JobTracker 它依然在运行，同时心跳中还携带着很多的信息，比如当前 map 任务完成的进度等信息。当 JobTracker 收到作业的最后一个任务完成信息时，便把该作业设置成“成功”。当 JobClient 查询状态时，它将得知任务已完成，便显示一条消息给用户；
8. 运行的 TaskTracker 从 HDFS 中获取运行所需要的资源，这些资源包括 MapReduce 程序打包的 JAR 文件、配置文件和客户端计算所得的输入划分等信息；
9. TaskTracker 获取资源后启动新的 JVM 虚拟机；
10. 运行每一个任务；

3 测试例子 1

3.1 测试例子 1 内容

下载气象数据集部分数据，写一个 Map-Reduce 作业，求每年的最低温度

3.2 运行代码

3.2.1 MinTemperature

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class MinTemperature {

    public static void main(String[] args) throws Exception {
        if(args.length != 2) {
            System.err.println("Usage: MinTemperature<input path> <output path>");
            System.exit(-1);
        }

        Job job = new Job();
        job.setJarByClass(MinTemperature.class);
        job.setJobName("Min temperature");
        FileInputFormat.addInputPath(job, new Path(args[0]));
```

```
        FileOutputFormat.setOutputPath(job, new Path(args[1]));
        job.setMapperClass(MinTemperatureMapper.class);
        job.setReducerClass(MinTemperatureReducer.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

3.2.2 MinTemperatureMapper

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class MinTemperatureMapper extends Mapper<LongWritable, Text, Text, IntWritable>{

    private static final int MISSING = 9999;

    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException {

        String line = value.toString();
        String year = line.substring(15, 19);

        int airTemperature;
        if(line.charAt(87) == '+') {
            airTemperature = Integer.parseInt(line.substring(88, 92));
        } else {
            airTemperature = Integer.parseInt(line.substring(87, 92));
        }

        String quality = line.substring(92, 93);
        if(airTemperature != MISSING && quality.matches("[01459]")) {
            context.write(new Text(year), new IntWritable(airTemperature));
        }
    }
}
```

3.2.3 MinTemperatureReducer

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;
```

```
public class MinTemperatureReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

    @Override
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws
IOException, InterruptedException {

        int minValue = Integer.MAX_VALUE;
        for(IntWritable value : values) {
            minValue = Math.min(minValue, value.get());
        }
        context.write(key, new IntWritable(minValue));
    }
}
```

3.3 实现过程

3.3.1 编写代码

进入/app/hadoop-1.1.2/myclass 目录，在该目录中建立 MinTemperature.java、MinTemperatureMapper.java 和 MinTemperatureReducer.java 代码文件，执行命令如下：

cd /app/hadoop-1.1.2/myclass/

vi MinTemperature.java

vi MinTemperatureMapper.java

vi MinTemperatureReducer.java

MinTemperature.java：

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class MinTemperature {

        public static void main(String[] args) throws Exception {
                if(args.length != 2) {
                        System.err.println("Usage: MinTemperature<input path> <output path>");
                        System.exit(-1);
                }

                Job job = new Job();
                job.setJarByClass(MinTemperature.class);
                job.setJobName("Min temperature");
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));
                job.setMapperClass(MinTemperatureMapper.class);
                job.setReducerClass(MinTemperatureReducer.class);
                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);
                System.exit(job.waitForCompletion(true) ? 0 : 1);
        }
}
```

MinTemperatureMapper.java：

```java
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class MinTemperatureMapper extends Mapper<LongWritable, Text, Text, IntWritable>{

        private static final int MISSING = 9999;

        @Override
        public void map(LongWritable key, Text value, Context context) throws IOException, Inter
ruptedException {

                String line = value.toString();
                String year = line.substring(15, 19);

                int airTemperature;
                if(line.charAt(87) == '+') {
                        airTemperature = Integer.parseInt(line.substring(88, 92));
                } else {
                        airTemperature = Integer.parseInt(line.substring(87, 92));
                }

                String quality = line.substring(92, 93);
                if(airTemperature != MISSING && quality.matches("[01459]")) {
                        context.write(new Text(year), new IntWritable(airTemperature));
                }
        }
}
```

MinTemperatureReducer.java:

```java
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class MinTemperatureReducer extends Reducer<Text, IntWritable, Text, IntWritable> {

        @Override
        public void reduce(Text key, Iterable<IntWritable> values, Context context) throws IOExc
eption, InterruptedException {

                int minValue = Integer.MAX_VALUE;
                for(IntWritable value : values) {
                        minValue = Math.min(minValue, value.get());
                }
                context.write(key, new IntWritable(minValue));
        }
}
```

3.3.2 编译代码

在/app/hadoop-1.1.2/myclass 目录中，使用如下命令对 java 代码进行编译，为保证编译成功，加入 classpath 变量，引入 hadoop-core-1.1.2.jar 包：

*javac -classpath ../hadoop-core-1.1.2.jar *.java*

```
[shiyanlou@b393a04554e1 myclass]$ javac -classpath ../hadoop-core-1.1.2.jar *.java
[shiyanlou@b393a04554e1 myclass]$ ll
total 52
-rw-rw-r-- 1 shiyanlou shiyanlou 1228 Jun  4 06:50 FileSystemCat.class
-rw-rw-r-- 1 shiyanlou shiyanlou  568 Jun  4 01:47 FileSystemCat.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1356 Jun  4 06:50 Hdfs2LocalFile.class
-rw-rw-r-- 1 shiyanlou shiyanlou  961 Jun  4 02:16 Hdfs2LocalFile.java
-rw-rw-r-- 1 shiyanlou shiyanlou  566 Jun  4 06:50 LocalFile2Hdfs$1.class
-rw-rw-r-- 1 shiyanlou shiyanlou 1465 Jun  4 06:50 LocalFile2Hdfs.class
-rw-rw-r-- 1 shiyanlou shiyanlou 1126 Jun  4 01:55 LocalFile2Hdfs.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1417 Jun  4 06:50 MinTemperature.class
-rw-rw-r-- 1 shiyanlou shiyanlou  953 Jun  4 06:48 MinTemperature.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1876 Jun  4 06:50 MinTemperatureMapper.class
-rw-rw-r-- 1 shiyanlou shiyanlou  901 Jun  4 06:49 MinTemperatureMapper.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1664 Jun  4 06:50 MinTemperatureReducer.class
-rw-rw-r-- 1 shiyanlou shiyanlou  552 Jun  4 06:50 MinTemperatureReducer.java
[shiyanlou@b393a04554e1 myclass]$
```

3.3.3 打包编译文件

把编译好 class 文件打包，否则在执行过程会发生错误。把打好的包移动到上级目录并删除编译

好的 class 文件：

jar cvf ./MinTemperature.jar ./Min.class*

*mv *.jar ..*

rm Min.class*

```
[shiyanlou@b393a04554e1 myclass]$ jar cvf ./MinTemperature.jar ./*.class
added manifest
adding: FileSystemCat.class(in = 1228) (out= 664)(deflated 45%)
adding: Hdfs2LocalFile.class(in = 1356) (out= 771)(deflated 43%)
adding: LocalFile2Hdfs$1.class(in = 566) (out= 376)(deflated 33%)
adding: LocalFile2Hdfs.class(in = 1465) (out= 840)(deflated 42%)
adding: MinTemperature.class(in = 1417) (out= 798)(deflated 43%)
adding: MinTemperatureMapper.class(in = 1876) (out= 803)(deflated 57%)
adding: MinTemperatureReducer.class(in = 1664) (out= 706)(deflated 57%)
[shiyanlou@b393a04554e1 myclass]$ ls
FileSystemCat.class     LocalFile2Hdfs.class  MinTemperatureMapper.class
FileSystemCat.java      LocalFile2Hdfs.java   MinTemperatureMapper.java
Hdfs2LocalFile.class    MinTemperature.class  MinTemperatureReducer.class
Hdfs2LocalFile.java     MinTemperature.jar    MinTemperatureReducer.java
LocalFile2Hdfs$1.class  MinTemperature.java
[shiyanlou@b393a04554e1 myclass]$ mv *.jar ..
[shiyanlou@b393a04554e1 myclass]$ rm Min*.class
[shiyanlou@b393a04554e1 myclass]$
```

3.3.4 解压气象数据并上传到 HDFS 中

把 NCDC 气象数据解压，并使用 zcat 命令把这些数据文件解压并合并到一个 temperature.txt 文件中

cd /home/shiyanlou

unzip temperature

cd temperature

*zcat *.gz > temperature.txt*

```
[shiyanlou@b393a04554e1 ~]$ pwd
/home/shiyanlou
[shiyanlou@b393a04554e1 ~]$ unzip temperature
Archive:  temperature.zip
   creating: temperature/
 extracting: temperature/010450-99999-1973.gz
 extracting: temperature/010470-99999-1973.gz
 extracting: temperature/010490-99999-1973.gz
 extracting: temperature/010510-99999-1973.gz
```

```
[shiyanlou@b393a04554e1 ~]$ cd temperature
[shiyanlou@b393a04554e1 temperature]$ zcat *.gz > temperature.txt
[shiyanlou@b393a04554e1 temperature]$ ls
010450-99999-1973.gz  295410-99999-1971.gz  380020-99999-1971.gz  724631-99999-1973.gz
010470-99999-1973.gz  295480-99999-1971.gz  380080-99999-1971.gz  724635-23067-1973.gz
010490-99999-1973.gz  295510-99999-1971.gz  380190-99999-1971.gz  724640-93058-1972.gz
010510-99999-1973.gz  295530-99999-1971.gz  380370-99999-1971.gz  724640-93058-1973.gz
010530-99999-1973.gz  295570-99999-1971.gz  723930-99999-1972.gz  874270-99999-1973.gz
010550-99999-1973.gz  295620-99999-1971.gz  724037-93728-1972.gz  874360-99999-1973.gz
010570-99999-1973.gz  295670-99999-1971.gz  724067-99999-1972.gz  874480-99999-1973.gz
080870-99999-1972.gz  375490-99999-1971.gz  724088-13707-1972.gz  874500-99999-1973.gz
082270-99999-1972.gz  375530-99999-1971.gz  724096-14706-1972.gz  874530-99999-1973.gz
083970-99999-1972.gz  375560-99999-1971.gz  724240-13807-1972.gz  874670-99999-1973.gz
085090-99999-1972.gz  375750-99999-1971.gz  724338-13802-1972.gz  999999-03813-1972.gz
106190-99999-1972.gz  376030-99999-1971.gz  724550-13947-1972.gz  999999-03815-1972.gz
106330-99999-1972.gz  376210-99999-1971.gz  724567-13921-1972.gz  999999-03820-1972.gz
106405-99999-1972.gz  376390-99999-1971.gz  724586-03919-1973.gz  999999-03822-1972.gz
106570-99999-1972.gz  379850-99999-1971.gz  724620-23061-1973.gz  999999-03840-1972.gz
295390-99999-1971.gz  379890-99999-1971.gz  724625-99999-1973.gz  999999-03853-1972.gz
295400-99999-1971.gz  380010-99999-1971.gz  724626-99999-1973.gz  temperature.txt
[shiyanlou@b393a04554e1 temperature]$
```

气象数据具体的下载地址为 ftp://ftp3.ncdc.noaa.gov/pub/data/noaa/ ，该数据包括 1900 年到现在所有年份的气象数据，大小大概有 70 多个 G，为了测试简单，我们这里选取一部分的数据进行测试。合并后把这个文件上传到 HDFS 文件系统的/class5/in 目录中：

hadoop fs -mkdir -p /class5/in

hadoop fs -copyFromLocal temperature.txt /class5/in

hadoop fs -ls /class5/in

```
[shiyanlou@b393a04554e1 temperature]$ hadoop fs -mkdir -p /class5/in
[shiyanlou@b393a04554e1 temperature]$ hadoop fs -copyFromLocal temperature.txt /class5/in
[shiyanlou@b393a04554e1 temperature]$ hadoop fs -ls /class5/in
Found 1 items
-rw-r--r--   1 shiyanlou supergroup   46337829 2015-06-04 07:12 /class5/in/temperature.txt
[shiyanlou@b393a04554e1 temperature]$
```

3.3.5 运行程序

以 jar 的方式启动 MapReduce 任务，执行输出目录为/class5/out：

cd /app/hadoop-1.1.2

hadoop jar MinTemperature.jar MinTemperature /class5/in/temperature.txt /class5/out

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ ls
bin                     hadoop-core-1.1.2.jar          ivy.xml             README.txt
build.xml               hadoop-examples-1.1.2.jar      lib                 sbin
c++                     hadoop-minicluster-1.1.2.jar   libexec             share
CHANGES.txt             hadoop-test-1.1.2.jar          LICENSE.txt         src
conf                    hadoop-tools-1.1.2.jar         logs                tmp
contrib                 hdfs                           MinTemperature.jar  webapps
hadoop-ant-1.1.2.jar    input                          myclass
hadoop-client-1.1.2.jar ivy                            NOTICE.txt
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar MinTemperature.jar MinTemperature /class5/in
/temperature.txt  /class5/out
15/06/04 07:15:07 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. A
pplications should implement Tool for the same.
15/06/04 07:15:07 INFO input.FileInputFormat: Total input paths to process : 1
15/06/04 07:15:07 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/04 07:15:07 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/04 07:15:08 INFO mapred.JobClient: Running job: job_201506040132_0002
15/06/04 07:15:09 INFO mapred.JobClient:  map 0% reduce 0%
15/06/04 07:15:18 INFO mapred.JobClient:  map 100% reduce 0%
15/06/04 07:15:26 INFO mapred.JobClient:  map 100% reduce 33%
15/06/04 07:15:28 INFO mapred.JobClient:  map 100% reduce 100%
15/06/04 07:15:29 INFO mapred.JobClient: Job complete: job_201506040132_0002
15/06/04 07:15:29 INFO mapred.JobClient: Counters: 29
```

3.3.6 查看结果

执行成功后 ,查看/class5/out 目录中是否存在运行结果 ,使用 cat 查看结果(温度需要除以 10):

hadoop fs -ls /class5/out

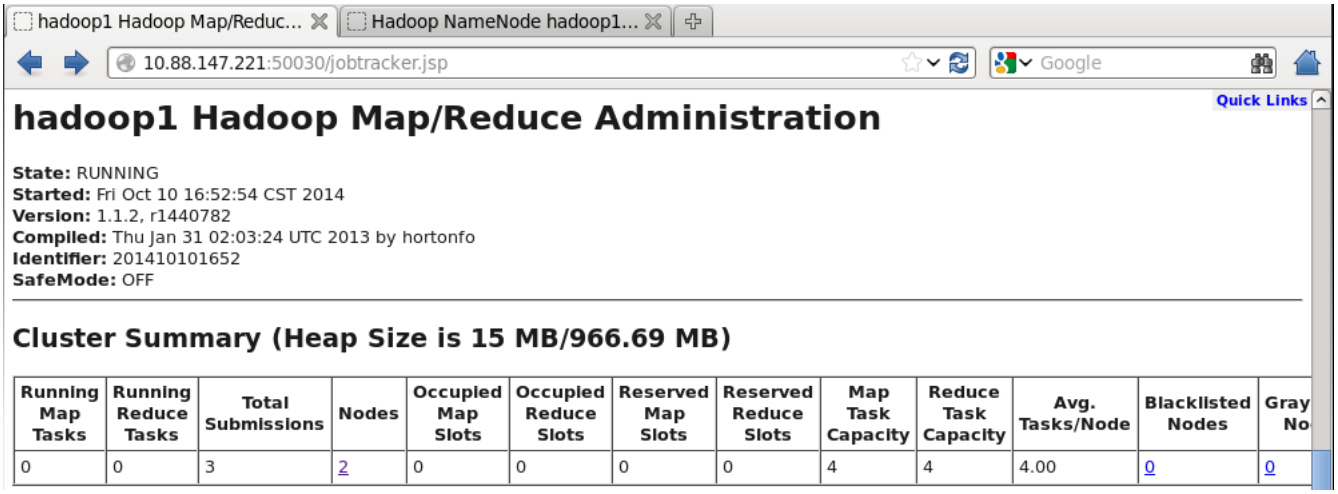
hadoop fs -cat /class5/out/part-r-00000

```
[shiyanlou@b393a04554e1 ~]$ hadoop fs -ls /class5/out
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-04 07:15 /class5/out/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-04 07:15 /class5/out/_logs
-rw-r--r--   1 shiyanlou supergroup         30 2015-06-04 07:15 /class5/out/part-r-00000
[shiyanlou@b393a04554e1 ~]$ hadoop fs -cat /class5/out/part-r-00000
1971    -461
1972    -267
1973    -390
[shiyanlou@b393a04554e1 ~]$
```

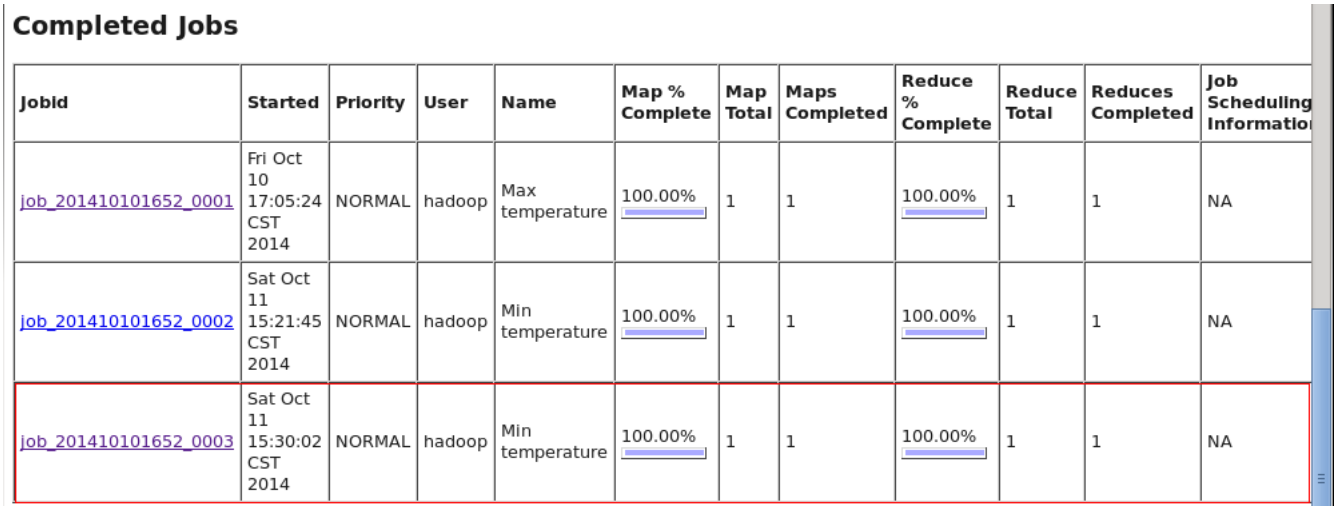
3.3.7 通过页面结果（由于实验楼环境是命令行界面，以下仅为说明运行过程和结果可以通过界面进行查看）

1. 查看 jobtracker.jsp

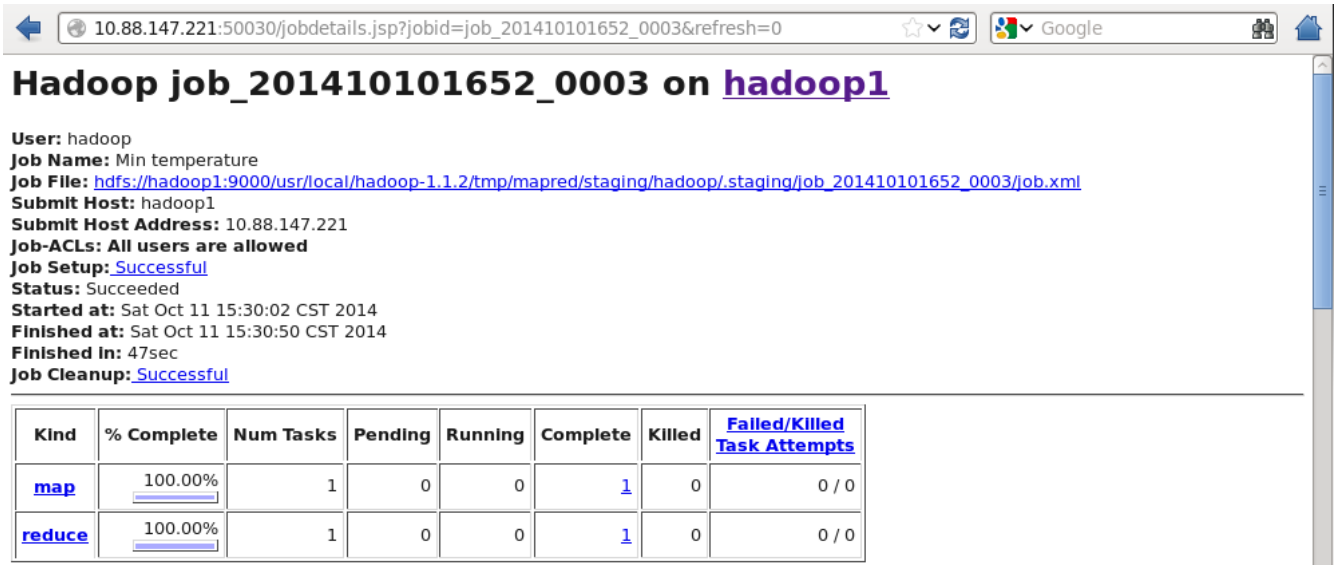
http://XX. XXX.XX.XXX:50030/jobtracker.jsp



查看已经完成的作业任务：

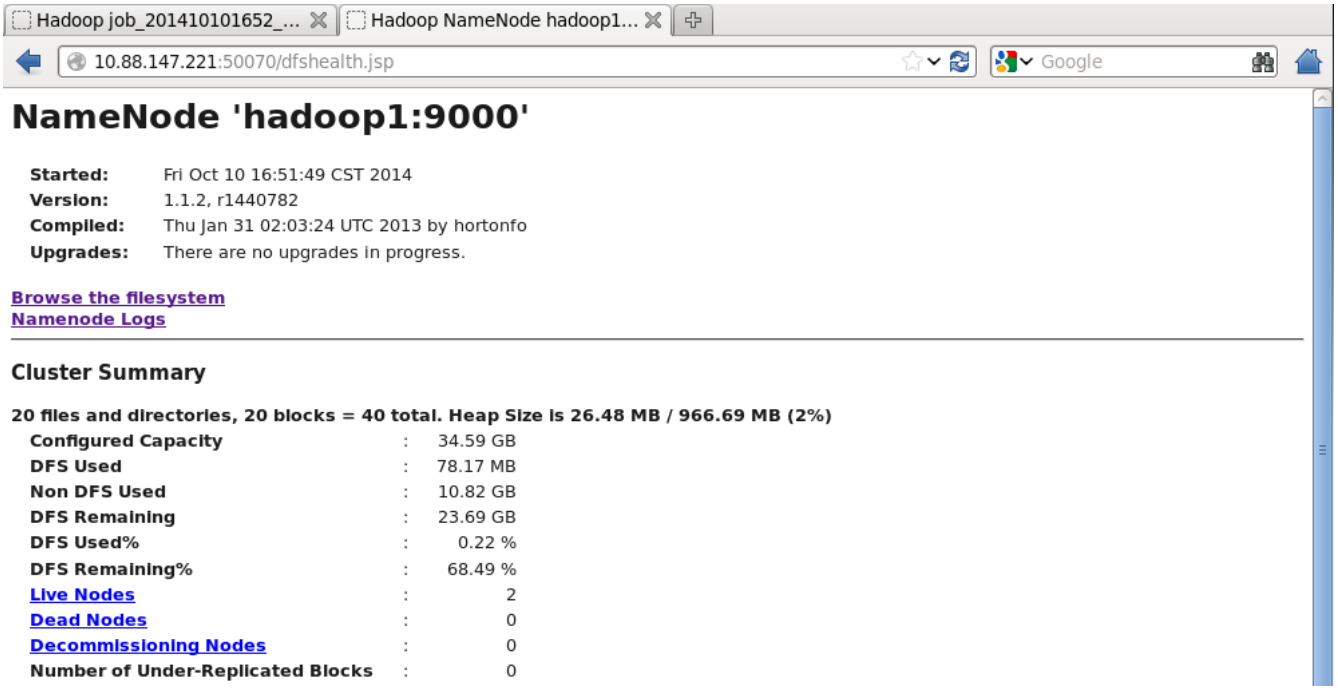


任务的详细信息：

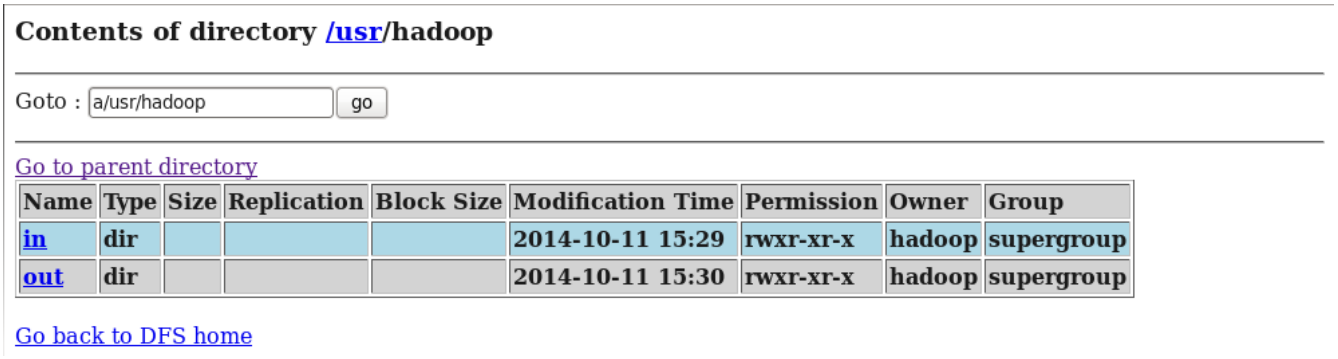


2. 查看 dfshealth.jsp

http://XX. XXX.XX.XXX:50070/dfshealth.jsp



分别查看 HDFS 文件系统和日志



4 测试例子 2

4.1 测试例子 2 内容

如果求温度的平均值，能使用 combiner 吗？有没有变通的方法？

4.2 回答

不能直接使用，因为求平均值和前面求最值存在差异，各局部最值的最值还是等于整体的最值的，但是对于平均值而言，各局部平均值的平均值将不再是整体的平均值了，所以不能直接用combiner。可以通过变通的办法使用 combiner 来计算平均值，即在 combiner 的键值对中不直接存储最后的平均值，而是存储所有值的和个数，最后在 reducer 输出时再用和除以个数得到平均值。

4.3 程序代码

4.3.1 AvgTemperature.java

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class AvgTemperature {

    public static void main(String[] args) throws Exception {

        if(args.length != 2) {
            System.out.println("Usage: AvgTemperatrue <input path><output path>");
            System.exit(-1);
        }

        Job job = new Job();
        job.setJarByClass(AvgTemperature.class);
        job.setJobName("Avg Temperature");
        FileInputFormat.addInputPath(job, new Path(args[0]));
        FileOutputFormat.setOutputPath(job, new Path(args[1]));

        job.setMapperClass(AvgTemperatureMapper.class);
        job.setCombinerClass(AvgTemperatureCombiner.class);
        job.setReducerClass(AvgTemperatureReducer.class);

        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(Text.class);

        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
```

```
        System.exit(job.waitForCompletion(true) ? 0 : 1);
    }
}
```

4.3.2 AvgTemperatureMapper.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class AvgTemperatureMapper extends Mapper<LongWritable, Text, Text, Text> {

    private static final int MISSING = 9999;

    @Override
    public void map(LongWritable key, Text value, Context context) throws IOException,
InterruptedException{

        String line = value.toString();
        String year = line.substring(15, 19);

        int airTemperature;
        if(line.charAt(87) == '+') {
            airTemperature = Integer.parseInt(line.substring(88, 92));
        } else {
            airTemperature =  Integer.parseInt(line.substring(87, 92));
        }

        String quality = line.substring(92, 93);
        if(airTemperature != MISSING && !quality.matches("[01459]")) {
            context.write(new Text(year), new Text(String.valueOf(airTemperature)));
        }
    }
}
```

4.3.3 AvgTemperatureCombiner.java

```
import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class AvgTemperatureCombiner extends Reducer<Text, Text, Text, Text>{

    @Override
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {
```

```
        double sumValue = 0;
        long numValue = 0;

        for(Text value : values) {
            sumValue += Double.parseDouble(value.toString());
            numValue ++;
        }

        context.write(key, new Text(String.valueOf(sumValue) + ',' +
String.valueOf(numValue)));
    }
}
```

4.3.4 AvgTemperatureReducer.java

```
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class AvgTemperatureReducer extends Reducer<Text, Text, Text, IntWritable>{

    @Override
    public void reduce(Text key, Iterable<Text> values, Context context) throws IOException,
InterruptedException {

        double sumValue = 0;
        long numValue = 0;
        int avgValue = 0;

        for(Text value : values) {
            String[] valueAll = value.toString().split(",");
            sumValue += Double.parseDouble(valueAll[0]);
            numValue += Integer.parseInt(valueAll[1]);
        }

        avgValue  = (int)(sumValue/numValue);
        context.write(key, new IntWritable(avgValue));
    }
}
```

4.4 实现过程

4.4.1 编写代码

进入/app/hadoop-1.1.2/myclass 目录，在该目录中建立 AvgTemperature.java、

AvgTemperatureMapper.java 、 AvgTemperatureCombiner.java 和 AvgTemperatureReducer.java 代码文件，代码内容为 4.3 所示，执行命令如下：

cd /app/hadoop-1.1.2/myclass/

vi AvgTemperature.java

vi AvgTemperatureMapper.java

vi AvgTemperatureCombiner.java

vi AvgTemperatureReducer.java

AvgTemperature.java：

```
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class AvgTemperature {

        public static void main(String[] args) throws Exception {

                if(args.length != 2) {
                        System.out.println("Usage: AvgTemperatrue <input path><output path>");
                        System.exit(-1);
                }

                Job job = new Job();
                job.setJarByClass(AvgTemperature.class);
                job.setJobName("Avg Temperature");
                FileInputFormat.addInputPath(job, new Path(args[0]));
                FileOutputFormat.setOutputPath(job, new Path(args[1]));

                job.setMapperClass(AvgTemperatureMapper.class);
                job.setCombinerClass(AvgTemperatureCombiner.class);
                job.setReducerClass(AvgTemperatureReducer.class);

                job.setMapOutputKeyClass(Text.class);
                job.setMapOutputValueClass(Text.class);

                job.setOutputKeyClass(Text.class);
                job.setOutputValueClass(IntWritable.class);

                System.exit(job.waitForCompletion(true) ? 0 : 1);
        }
}
```

AvgTemperatureMapper.java：

```java
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Mapper;

public class AvgTemperatureMapper extends Mapper<LongWritable, Text, Text, Text> {

	private static final int MISSING = 9999;

	@Override
	public void map(LongWritable key, Text value, Context context) throws IOException, Int
erruptedException{

		String line = value.toString();
		String year = line.substring(15, 19);

		int airTemperature;
		if(line.charAt(87) == '+') {
			airTemperature = Integer.parseInt(line.substring(88, 92));
		} else {
			airTemperature =  Integer.parseInt(line.substring(87, 92));
		}

		String quality = line.substring(92, 93);
		if(airTemperature != MISSING && !quality.matches("[01459]")) {
			context.write(new Text(year), new Text(String.valueOf(airTemperature))
);
		}
	}
}
```

AvgTemperatureCombiner.java：

```java
import java.io.IOException;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class AvgTemperatureCombiner extends Reducer<Text, Text, Text, Text>{

	@Override
	public void reduce(Text key, Iterable<Text> values, Context context) throws IOExceptio
n, InterruptedException {

		double sumValue = 0;
		long numValue = 0;

		for(Text value : values) {
			sumValue += Double.parseDouble(value.toString());
			numValue ++;
		}

		context.write(key, new Text(String.valueOf(sumValue) + ',' + String.valueOf(nu
mValue)));
	}
}
```

AvgTemperatureReducer.java:

```java
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Reducer;

public class AvgTemperatureReducer extends Reducer<Text, Text, Text, IntWritable>{

	@Override
	public void reduce(Text key, Iterable<Text> values, Context context) throws IOExceptio
n, InterruptedException {

		double sumValue = 0;
		long numValue = 0;
		int avgValue = 0;

		for(Text value : values) {
			String[] valueAll = value.toString().split(",");
			sumValue += Double.parseDouble(valueAll[0]);
			numValue += Integer.parseInt(valueAll[1]);
		}

		avgValue  = (int)(sumValue/numValue);
		context.write(key, new IntWritable(avgValue));
	}
}
```

4.4.2 编译代码

在/app/hadoop-1.1.2/myclass 目录中 ,使用如下命令对 java 代码进行编译 ,为保证编译成功，加入 classpath 变量，引入 hadoop-core-1.1.2.jar 包：

javac -classpath ../hadoop-core-1.1.2.jar Avg.java*

```
[shiyanlou@b393a04554e1 myclass]$ javac -classpath ../hadoop-core-1.1.2.jar Avg*.java
[shiyanlou@b393a04554e1 myclass]$ ll
total 56
-rw-rw-r-- 1 shiyanlou shiyanlou 1575 Jun  4 07:28 AvgTemperature.class
-rw-rw-r-- 1 shiyanlou shiyanlou 1947 Jun  4 07:28 AvgTemperatureCombiner.class
-rw-rw-r-- 1 shiyanlou shiyanlou  561 Jun  4 07:27 AvgTemperatureCombiner.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1105 Jun  4 07:26 AvgTemperature.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1854 Jun  4 07:28 AvgTemperatureMapper.class
-rw-rw-r-- 1 shiyanlou shiyanlou  904 Jun  4 07:27 AvgTemperatureMapper.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1932 Jun  4 07:28 AvgTemperatureReducer.class
-rw-rw-r-- 1 shiyanlou shiyanlou  703 Jun  4 07:28 AvgTemperatureReducer.java
-rw-rw-r-- 1 shiyanlou shiyanlou  568 Jun  4 01:47 FileSystemCat.java
-rw-rw-r-- 1 shiyanlou shiyanlou  961 Jun  4 02:16 Hdfs2LocalFile.java
-rw-rw-r-- 1 shiyanlou shiyanlou 1126 Jun  4 01:55 LocalFile2Hdfs.java
-rw-rw-r-- 1 shiyanlou shiyanlou  953 Jun  4 06:48 MinTemperature.java
-rw-rw-r-- 1 shiyanlou shiyanlou  901 Jun  4 06:49 MinTemperatureMapper.java
-rw-rw-r-- 1 shiyanlou shiyanlou  552 Jun  4 06:50 MinTemperatureReducer.java
[shiyanlou@b393a04554e1 myclass]$
```

4.4.3 打包编译文件

把编译好 class 文件打包，否则在执行过程会发生错误。把打好的包移动到上级目录并删除编译好的 class 文件：

jar cvf ./AvgTemperature.jar ./Avg.class*

ls

*mv *.jar ..*

rm Avg.class*

```
[shiyanlou@b393a04554e1 myclass]$ jar cvf ./AvgTemperature.jar ./Avg*.class
added manifest
adding: AvgTemperature.class(in = 1575) (out= 872)(deflated 44%)
adding: AvgTemperatureCombiner.class(in = 1947) (out= 823)(deflated 57%)
adding: AvgTemperatureMapper.class(in = 1854) (out= 810)(deflated 56%)
adding: AvgTemperatureReducer.class(in = 1932) (out= 831)(deflated 56%)
[shiyanlou@b393a04554e1 myclass]$ ls
AvgTemperature.class          AvgTemperatureMapper.class    Hdfs2LocalFile.java
AvgTemperatureCombiner.class  AvgTemperatureMapper.java     LocalFile2Hdfs.java
AvgTemperatureCombiner.java   AvgTemperatureReducer.class   MinTemperature.java
AvgTemperature.jar            AvgTemperatureReducer.java    MinTemperatureMapper.java
AvgTemperature.java           FileSystemCat.java            MinTemperatureReducer.java
[shiyanlou@b393a04554e1 myclass]$ mv *.jar ..
[shiyanlou@b393a04554e1 myclass]$ rm Avg*.class
[shiyanlou@b393a04554e1 myclass]$
```

4.4.4 运行程序

数据使用作业 2 求每年最低温度的气象数据 数据在 HDFS 位置为/class5/in/temperature.txt，以 jar 的方式启动 MapReduce 任务，执行输出目录为/class5/out2：

cd /app/hadoop-1.1.2

hadoop jar AvgTemperature.jar AvgTemperature /class5/in/temperature.txt /class5/out2

```
[shiyanlou@b393a04554e1 ~]$ cd /app/hadoop-1.1.2
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ ls
AvgTemperature.jar  hadoop-ant-1.1.2.jar          hdfs        logs                 src
bin                 hadoop-client-1.1.2.jar       input       MinTemperature.jar   tmp
build.xml           hadoop-core-1.1.2.jar         ivy         myclass              webapps
c++                 hadoop-examples-1.1.2.jar     ivy.xml     NOTICE.txt
CHANGES.txt         hadoop-minicluster-1.1.2.jar  lib         README.txt
conf                hadoop-test-1.1.2.jar         libexec     sbin
contrib             hadoop-tools-1.1.2.jar        LICENSE.txt share
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop jar AvgTemperature.jar AvgTemperature /class5/in
/temperature.txt /class5/out2
15/06/04 07:32:09 WARN mapred.JobClient: Use GenericOptionsParser for parsing the arguments. A
pplications should implement Tool for the same.
15/06/04 07:32:09 INFO input.FileInputFormat: Total input paths to process : 1
15/06/04 07:32:09 INFO util.NativeCodeLoader: Loaded the native-hadoop library
15/06/04 07:32:09 WARN snappy.LoadSnappy: Snappy native library not loaded
15/06/04 07:32:10 INFO mapred.JobClient: Running job: job_201506040132_0003
15/06/04 07:32:11 INFO mapred.JobClient:  map 0% reduce 0%
15/06/04 07:32:18 INFO mapred.JobClient:  map 100% reduce 0%
15/06/04 07:32:27 INFO mapred.JobClient:  map 100% reduce 33%
15/06/04 07:32:28 INFO mapred.JobClient:  map 100% reduce 100%
15/06/04 07:32:29 INFO mapred.JobClient: Job complete: job_201506040132_0003
15/06/04 07:32:29 INFO mapred.JobClient: Counters: 29
```

4.4.5 查看结果

执行成功后，查看/class5/out2 目录中是否存在运行结果，使用 cat 查看结果（温度需要除以 10）：

hadoop fs -ls /class5/out2

hadoop fs -cat /class5/out2/part-r-00000

```
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -ls /class5/out2
Found 3 items
-rw-r--r--   1 shiyanlou supergroup          0 2015-06-04 07:32 /class5/out2/_SUCCESS
drwxr-xr-x   - shiyanlou supergroup          0 2015-06-04 07:32 /class5/out2/_logs
-rw-r--r--   1 shiyanlou supergroup         26 2015-06-04 07:32 /class5/out2/part-r-00000
[shiyanlou@b393a04554e1 hadoop-1.1.2]$ hadoop fs -cat /class5/out2/part-r-00000
1971    69
1972    147
1973    176
[shiyanlou@b393a04554e1 hadoop-1.1.2]$
```