REPORT

Cloud Computing Project 1

Transformation of data during the computations, data type of key, value:

We have used the Approach B as suggested by Saliya Ekanayake to include a delimiter tag after every 10 inputs in the input file. We first pre-processed the input file to get 10 elements in each line separated by “,”. As the DataInput by itself reads one line at a time, So according to the input which we are giving to the file, each line will contain 10 comma separated elements which are then split using String.split(“,”) and those values are stored as an array of elements.

We computed the partial statistical values: partialSum, partialSquareSum, partialMin, partialMax, partialCount using the array of 10 elements found in each line in the Map phase which are then written to a CompositeWritable object. This object is then passed to the reducer.
In the Reduce phase, we iterate over the CompositeWritable object to compute the global statistical values: globalSum, globalSquareSum, globalMin, globalMax, globalCount, globalAverage and globalSD. These values are then written to the output file.

As the input to the Mapper, we have used <LongWritable, Text> as a <key, value> pair and <Text, CompositeWritable> as a key value pair is used as the output for the Mapper.
Which makes, the <key, value> pair <Text, CompositeWritable> as an input to the reducer. The output of the Reducer is the <key, value> pair of <Text, DoubleWritable>.

The data structure used to transfer between Map and Reduce phases:

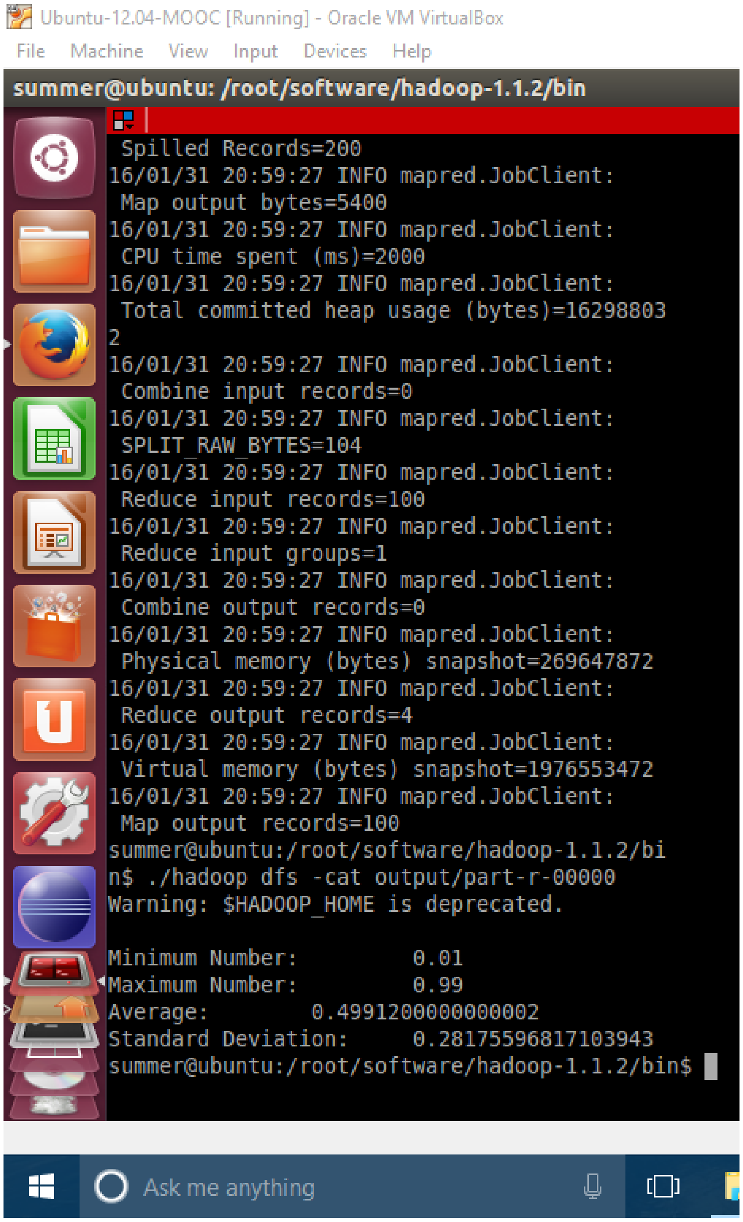
The CustomWritable class is created to transfer the data between the Map and Reduce phases. The CustomWritable class contains 5 double values namely: partialSum, partialSquareSum, partialMin, partialMax, partialCount. These 5 values will be computed in the Map phase. The CustomWritable object is then output to the Reducer as the “value” in the <key, value> pair. The reducer then iterates over the CustomWritable objects to compute globalSum, globalSquareSum, globalMin, globalMax, globalCount, globalAverage and globalSD which are the various required statistical values.

How the data flow happens through disk and memory during the computation:

We first pre-processed the input text file to include 10 comma separated values per line. This processed input file is then stored in the input folder on the local disk. This input file is then copied to the Hadoop Distributed File System. We then read this text file line by line and the values are then stored in a double array in the memory. We then perform computations and calculate the partial values in the Map phase for each of these 10 elements. We then calculate the global values using each of these partial values in the reduce phase. Then these values are written to the output folder in the Hadoop Distributed File System.

Note: We have also included the processed input file which was used in the program and the code used to generate it

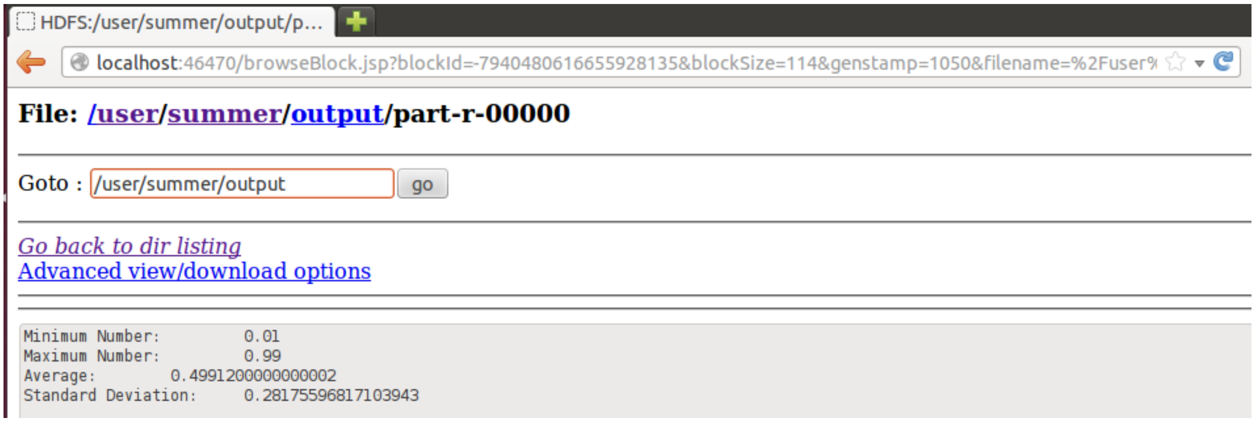
Output:



Ubuntu-12.04-MOOC [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
summer@ubuntu: /root/software/hadoop-1.1.2/bin
Ask me anything

```
 Spilled Records=200
16/01/31 20:59:27 INFO mapred.JobClient:
 Map output bytes=5400
16/01/31 20:59:27 INFO mapred.JobClient:
 CPU time spent (ms)=2000
16/01/31 20:59:27 INFO mapred.JobClient:
 Total committed heap usage (bytes)=16298803
2
16/01/31 20:59:27 INFO mapred.JobClient:
 Combine input records=0
16/01/31 20:59:27 INFO mapred.JobClient:
 SPLIT_RAW_BYTES=104
16/01/31 20:59:27 INFO mapred.JobClient:
 Reduce input records=100
16/01/31 20:59:27 INFO mapred.JobClient:
 Reduce input groups=1
16/01/31 20:59:27 INFO mapred.JobClient:
 Combine output records=0
16/01/31 20:59:27 INFO mapred.JobClient:
 Physical memory (bytes) snapshot=269647872
16/01/31 20:59:27 INFO mapred.JobClient:
 Reduce output records=4
16/01/31 20:59:27 INFO mapred.JobClient:
 Virtual memory (bytes) snapshot=1976553472
16/01/31 20:59:27 INFO mapred.JobClient:
 Map output records=100
summer@ubuntu:/root/software/hadoop-1.1.2/bi
n$ ./hadoop dfs -cat output/part-r-00000
Warning: $HADOOP_HOME is deprecated.

Minimum Number:         0.01
Maximum Number:         0.99
Average:        0.4991200000000002
Standard Deviation:     0.28175596817103943
summer@ubuntu:/root/software/hadoop-1.1.2/bin$ 
```



HDFS:/user/summer/output/p...
localhost:46470/browseBlock.jsp?blockId=-7940480616655928135&blockSize=114&genstamp=1050&filename=%2Fuser%

File: /user/summer/output/part-r-00000

Goto : /user/summer/output go

Go back to dir listing
Advanced view/download options

```
Minimum Number:         0.01
Maximum Number:         0.99
Average:        0.4991200000000002
Standard Deviation:     0.28175596817103943
```