TeraSort - (Distributed Sort using MapReduce)

1. Map phase:

InputReader, map, partition and combiner

1.1 Line contains a data

A single record is of 100 bytes binary

1.2 Input to map() function:

<byteoffset, element>

Output of map() function:

<element, blank>

Note that the value of input becomes key for output.

1.3 Partitioner

Perform a quick sort of data set and partial ordering such a way that data is partitioned into different partitions based on the range and not hash. I.e. a to b is partition id 1, b+1 to c is in partition id 2, c+1 to d is in partition id 3,... , y+1 to z is in partition 25. Just an analogy with the alphabet for easy grasping.

1.4 Combiner
N/A

2. Reduce Phase

2.1 Sort and shuffle
It’s very much a merge sort. But part of the framework itself.

2.2 Reduce

Framework to make sure that partition1 has been worked out by reducer1 which just forwards input as output. (Identity function)

The total ordering of execution of reducers confirms that final resultset is completely/globally sorted. It is also verified by TeraValidate for internal boundaries.

Check out the complete program package for better understanding of logic and implementation.

Data structure two level “Trie” is in use.

https://github.com/apache/hadoop-common/tree/trunk/hadoop-mapreduce-project/hadoop-mapreduce-examples/src/main/java/org/apache/hadoop/examples/terasort

```
10 billion records having each 100 bytes of binary information totaling 1 TB
total data size.
```