Package org.apache.hadoop.examples.terasort

This package consists of 3 map/reduce applications for Hadoop to compete in the annual terabyte sort competition.
This package consists of 3 map/reduce applications for Hadoop to compete in the annual terabyte sort competition.

- **TeraGen** is a map/reduce program to generate the data.
- **TeraSort** samples the input data and uses map/reduce to sort the data into a total order.
- **TeraValidate** is a map/reduce program that validates the output is sorted.

TeraGen generates output data that is byte for byte equivalent to the C version including the newlines and specific keys. It divides the desired number of rows by the desired number of tasks and assigns ranges of rows to each map. The map jumps the random number generator to the correct value for the first row and generates the following rows.

TeraSort is a standard map/reduce sort, except for a custom partitioner that uses a sorted list of *N-1* sampled keys that define the key range for each reduce. In particular, all keys such that *sample[i-1] <= key <*

sample[i] are sent to reduce *i*. This guarantees that the output of reduce *i* are all less than the output of reduce *i+1*. To speed up the partitioning, the partitioner builds a two level trie that quickly indexes into the list of sample keys based on the first two bytes of the key. TeraSort generates the sample keys by sampling the input before the job is submitted and writing the list of keys into HDFS. The input and output format, which are used by all 3 applications, read and write the text files in the right format. The output of the reduce has replication set to 1, instead of the default 3, because the contest does not require the output data be replicated on to multiple nodes.

TeraValidate ensures that the output is globally sorted. It creates one map per a file in the output directory and each map ensures that each key is less than or equal to the previous one. The map also generates records with the first and last keys of the file and the reduce ensures that the first key of file *i* is greater that the last key of file *i-1*. Any problems are reported as output of the reduce with the keys that are out of order.

In May 2008, Owen O'Malley ran this code on a 910 node cluster and sorted the 10 billion records (1 TB) in 209 seconds (3.48 minutes) to win the annual general purpose (daytona) terabyte sort benchmark.

The cluster statistics were:

- 910 nodes
- 4 dual core Xeons @ 2.0ghz per a node
- 4 SATA disks per a node
- 8G RAM per a node
- 1 gigabit ethernet on each node
- 40 nodes per a rack
- 8 gigabit ethernet uplinks from each rack to the core
- Red Hat Enterprise Linux Server Release 5.1 (kernel 2.6.18)
- Sun Java JDK 1.6.0_05-b13

The test was on Hadoop trunk (pre-0.18) patched with HADOOP-3443 and HADOOP-3446, which were required to remove intermediate writes to disk. TeraGen used 1800 tasks to generate a total of 10 billion rows in HDFS, with a block size of 1024 MB. TeraSort was configured with 1800 maps and 1800 reduces, and *mapreduce.task.io.sort.mb*, *mapreduce.task.io.sort.factor*, *fs.inmemory.size.mb*, and task heap size sufficient that transient data was never spilled to disk, other at the end of the map. The sampler looked at 100,000 keys to determine the reduce boundaries, which lead to imperfect balancing with reduce outputs ranging from 337 MB to 872 MB.