HadoopVsGridGain

Comparing GridGain to Hadoop

This comparison is based on my knowledge of Hadoop and the JavaDocs for GridGain 2.0.2. With respect to names, Hadoop jobs are broken down into many tasks, while GridGain reverses that. For this discussion, I'll stick to the Hadoop meanings to avoid confusion.

The primary difference is that Hadoop is designed to work with large data sets (100's of TB in a single job) and GridGain is not. GridGain's jobs have a single reducer and it is given all of the values in a java.util.List. Therefore, the GridGain's jobs are limited to what can fit in a single jvm's heap.

In GridGain's system each map and reduce returns a single value. To support map/reduce like semantics, each value would need to be a list. In Hadoop, each map and reduce may generate zero or more key/value pairs that are serialized as they are generated.

GridGain provides only distributed computation support and doesn't have a distributed file system. Hadoop provides HDFS that provides reliable scalable store to 1000's of nodes and PB of data.

GridGain's framework does not sort the data between the maps and the reduce. However, since it is passed as a java.util.List to the reduce, it is pretty easy for the application to sort it as desired. Hadoop provides an automatic distributed sort of the data between the maps and reduces.

GridGain does not support combiners, or counters. Combiners are an optional pass that reduces the values out of each map to shrink the amount of data that needs to be shuffled. Counters are user and system defined events that are counted and are used to track the progress of the job as it runs.

GridGain uses java.io serialization, while Hadoop uses a much more flexible serialization, which can use either java.io or user-defined serialization.

Hadoop provides a web interface to track the job's progress and I don't see any similar functionality in GridGain.

GridGain's model is more flexible in that the job can create arbitrary tasks, each with different input and code to run. The application gets a callback when each task finishes and can easily cancel the rest of the job's tasks.

GridGain doesn't have any support for non-Java applications, while Hadoop supports both C++ and text-based applications.

GridGain's map method returns a Map<Task, Node>, which allows it to do task locality. However, the locality is not a suggestion, but an order. To handle over-subscribed nodes, GridGain supports application collision strategies which can accept, queue, or reject tasks sent to the node. Hadoop's InputSplits define a list of nodes that it would prefer to run on, but it will run on the first otherwise idle node.

Support is available for both Hadoop and GridGain. Hadoop by the open source community and the companies listed on the support page. GridGain is supported by GridGain.