Concurrency & Parallel Programming

Hadoop

Auteurs: Tom Peerdeman & René Aparicio Saez

Datum: 03-12-2012

1 What is Hadoop

Hadoop is a piece of software software that allows certain programs to be split in smaller parts in such a way that multiple computations can be done on multiple nodes at the same times. In that way the time needed to compute the code decreases. It is however necessary that the parts have no dependencies of one another.

1.1 Hadoop's origin

Hadoop was created by looking at the MapReduce and Google File System papers made by Google. The essence of MapReduce is used by Hadoop. The master node takes input. This input is then divided in smaller parts and distributed onto worker (or slave) nodes. These worker nodes do all the computing. The workers cannot communicate with eachother, this is why there must be no dependencies what so ever in the given parts. When the workers are done they send their results back to their master. The master waits for all the answers and combines all the results to form an output.

1.2 Scalable

Hadoop is scalable. This means that you can assign more and more workers to the master node and receive faster computations because of this (this will eventually not happen anymore, because there isn'y enough work to give to all the nodes).

1.3 Architecture

Hadoop uses a special Distributed File System called the Hadoop Distributed File System or HDFS. HDFS uses location awareness to tell the master where and what each worker is doing. This way, work that must be done can be scheduled more efficiently. Also, if an exception is caused on a worker, the master will know what happened. If an exception occured it resends the job to be computed again.
To know all this, a master node consists of four parts: a JobTracker, a TaskTracker, a NameNode and a DataNode. The worker however, only has a TaskTracker and a DataNode. NameNodes contain information about where data is kept, but does not store data itself. Data is kept inside DataNodes. These connect to DataNodes to tell the DataNodes where the data is. The JobTracker talks to the NameNode to see what data can be sent. If data is found it makes contact with a TaskTracker of an open node and sends the information of the NameNode to this node. Preferably it sends the task to an empty slot of a node on the same server, else the task is send to any empty slot in the same rack of machines. If the TaskTracker accepted the node it either tells the JobTracker it is completely full or that it still has room for more work. These TaskTrackers then start sending messages back, telling the JobTracker that everything is ok, or if something went wrong that they failed at their job. If the latter happened, JobTracker can send out the task to a new node again. When everything happened according to plan, the JobTracker changes the status of the worker from occupied to free.