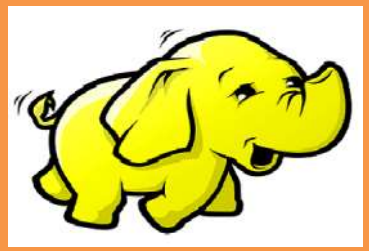
SNS COLLEGE OF TECHNOLOGY

Hadoop Library from Apache

DEPARTMENT OF COMPUTER APPLICATIONS
SNS COLLEGE OF TECHNOLOGY
COIMBATORE – 64035

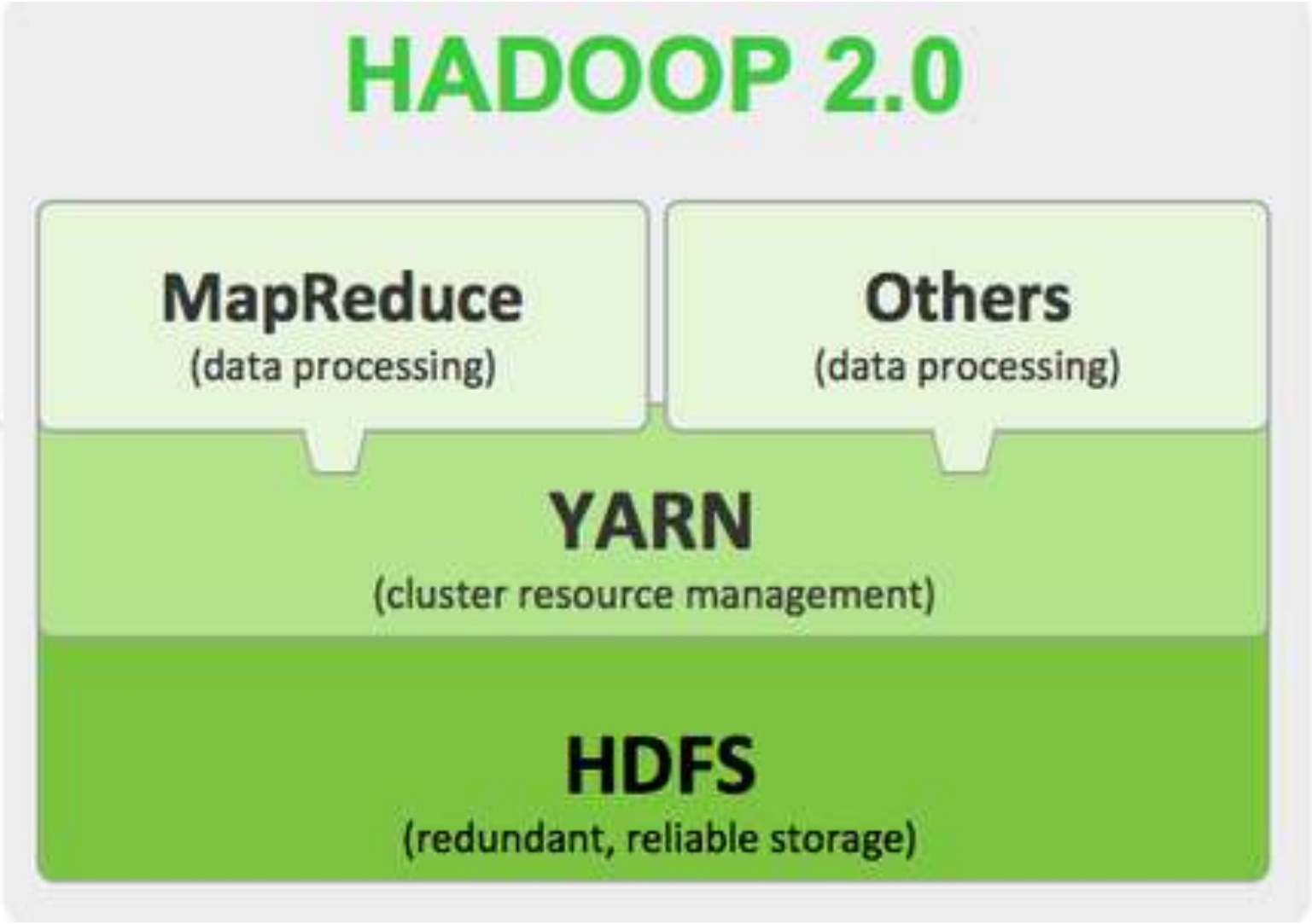


HADOOP

Top level, open-source implementation of frame works for reliable, scalable, distributed computing and data storage.

Why we need?

- Need to process huge datasets on large clusters of computers
- Very expensive to build reliability into each application
- Nodes fail every day
  - Failure is expected, rather than exceptional
  - The number of nodes in a cluster is not constant
- Need a common infrastructure
  - Efficient, reliable, easy to use
  - Open Source, Apache Licence


HADOOP 2.0
MapReduce
(data processing)
Others
(data processing)
YARN
(cluster resource management)
HDFS
(redundant, reliable storage)

Hadoop Distributed File Syste

A distributed file system inspired by GFS that organizes files and stores their data on a distributed computing system

Distinguished characteristics

HDFS Fault Tolerance:

- Hadoop is designed to be deployed on low-cost /commodity hardware.

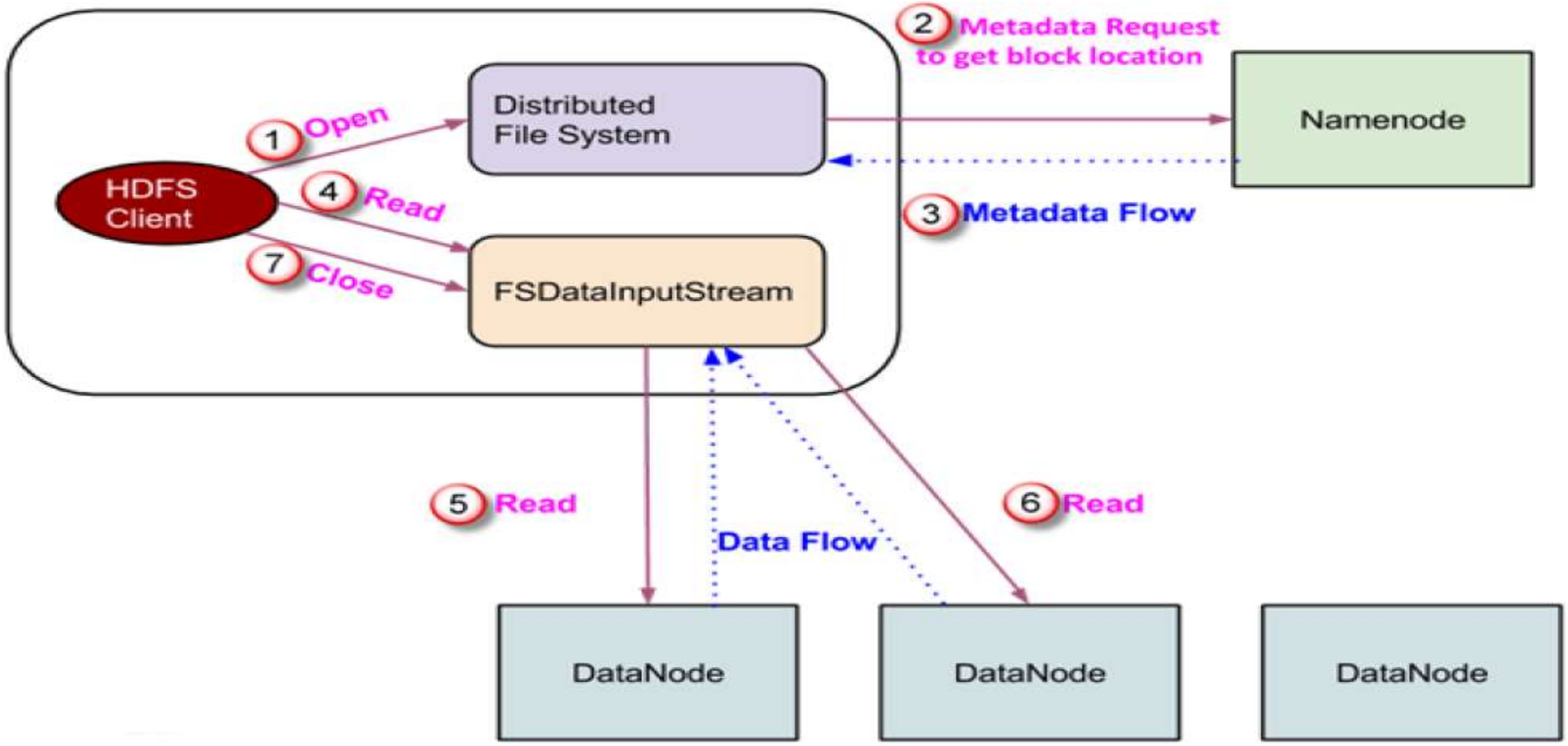
HDFS High-Throughput Access to Large Data Sets (Files):

Individual large data files are broken into large blocks (e.g., 64 MB) to allow HDFS to decrease the amount of metadata storage required per file

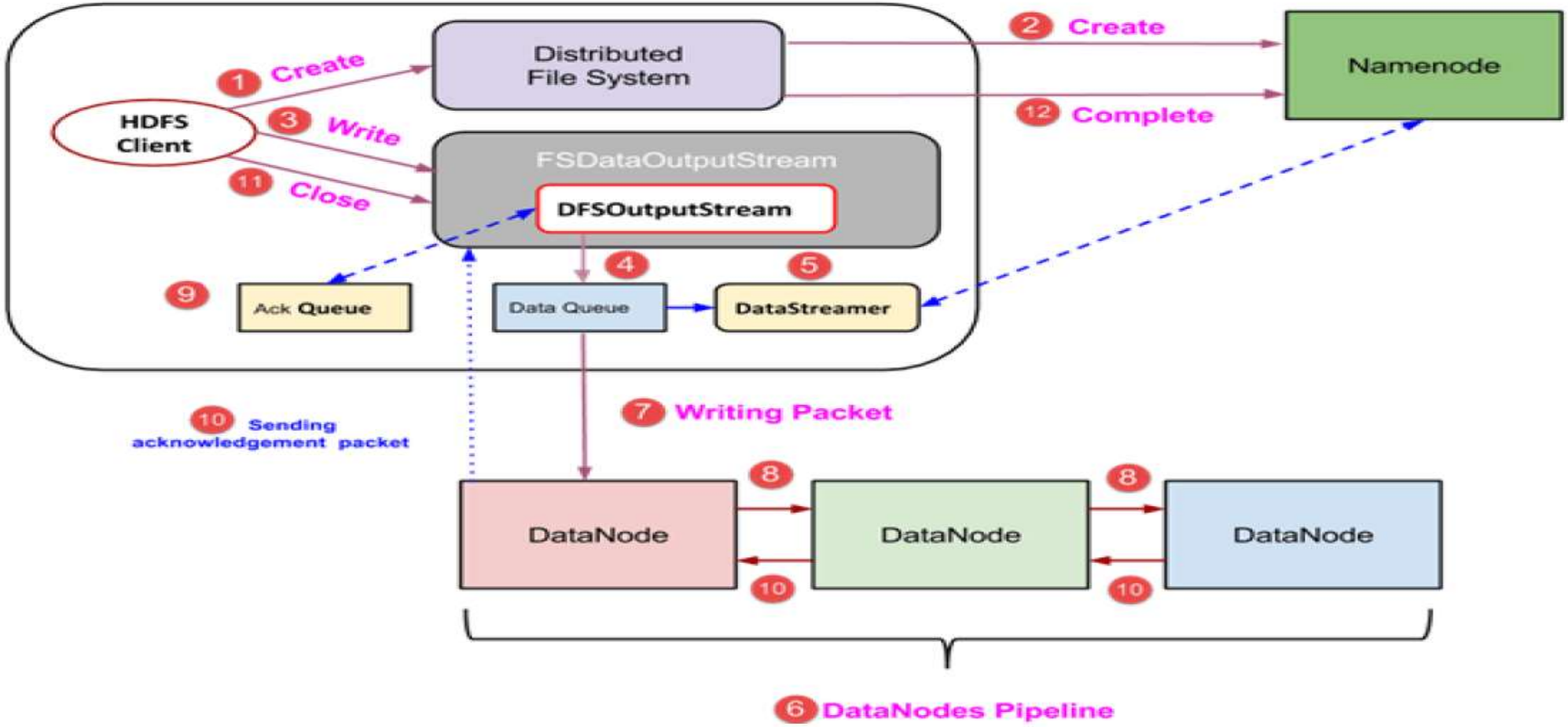
Advantages:

- list of blocks per file will shrink as the size of individual blocks increases
- by keeping large amounts of data sequentially within a block, HDFS provides fast streaming reads of data

Read Operation



Write Operation



HDFS- Fault tolerance

- The following file system issues are considered to fulfill reliability requirements
- **Block replication:** HDFS stores a file as a set of blocks and each block is replicated and distributed across the whole cluster
- **Replica placement:** storing replicas on different nodes (DataNodes) located in different racks across the whole cluster provides more reliability
- Default replication factor of three, HDFS stores one replica in the same node the original data is stored, one replica on a different node but in the same rack, and one replica on a different node in a different rack to provide three copies of the data
- **Heartbeat and Blockreport messages**: periodic messages sent to the NameNode by each DataNode in a cluster.

HDFS- Architecture

Master/slave architecture containing a single NameNode as the master and a number of DataNodes as workers (slaves)

- HDFS splits the file into fixed-size blocks (e.g., 64 MB) and stores them on DataNodes
- The mapping of blocks to DataNodes is determined by NameNode
- NameNode (master) also manages the file system's metadata and namespace
- Each DataNode is responsible for storing and retrieving its file blocks

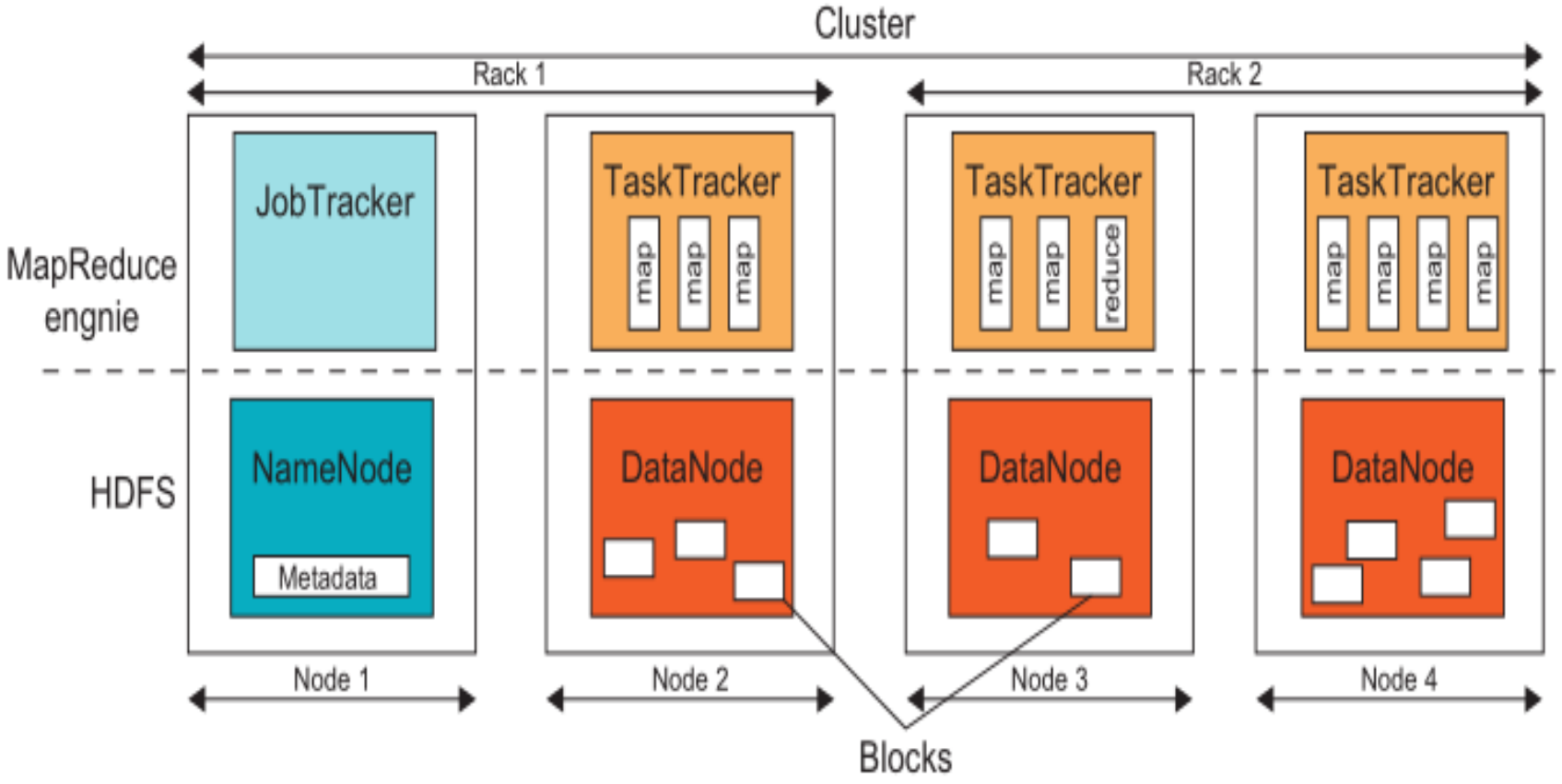
MapReduce- Architecture

Master/slave architecture consisting of a single JobTracker as the master and a number of TaskTrackers as the slaves

- The JobTracker manages the MapReduce job over a cluster and is responsible for monitoring jobs and assigning tasks to TaskTrackers
- TaskTracker manages the execution of the map and/or reduce tasks on a node in the cluster
- Each TaskTracker node has a number of simultaneous execution slots, each executing either a map / reduce task. Slots are nothing but no. of simultaneous threads supported by CPUs of the TaskTracker node

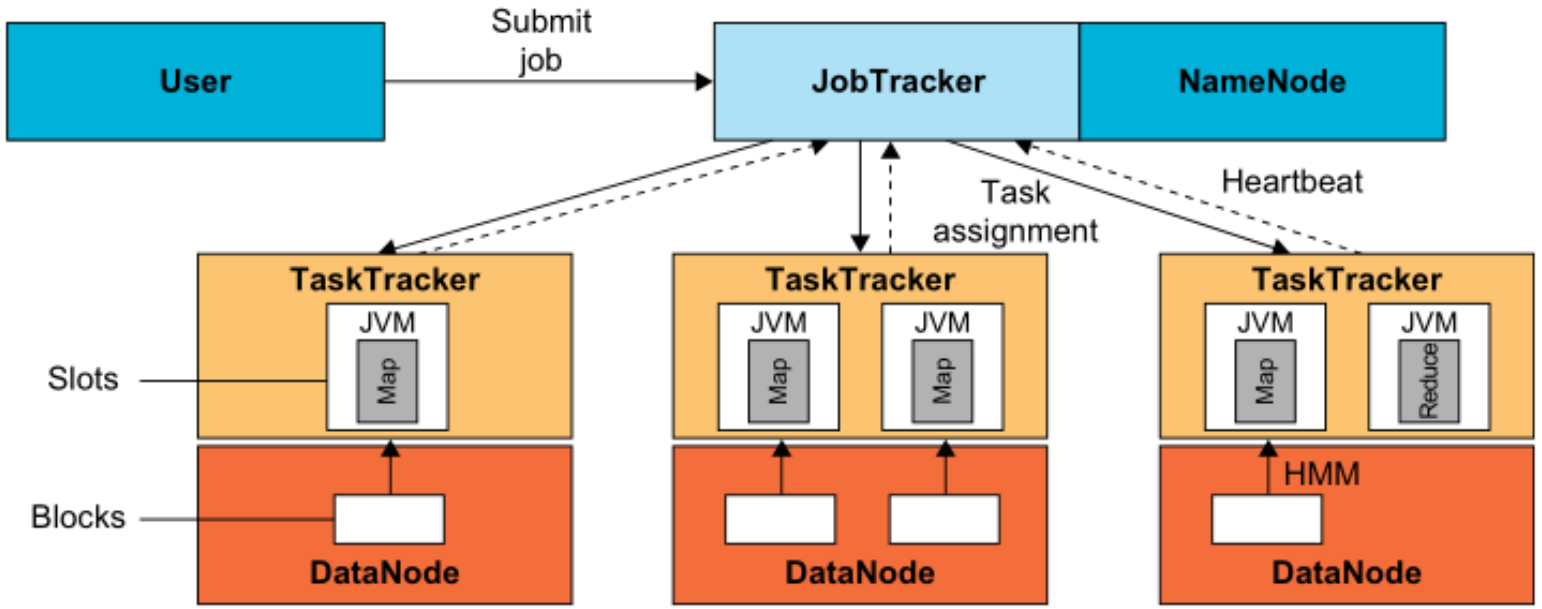
MapReduce- Architecture

Figure shows the MapReduce engine architecture cooperating with HDFS



Running a Job in Hadoop

- Data flow starts by calling the runJob(conf) function inside a user program running on the user node
- conf is an object containing some tuning parameters



Running a Job in Hadoop

Job Submission:

- User node submits each job to JobTracker node that situated in a different node within the cluster through
- A user node asks for a new job ID from JobTracker and computes input file splits
- user node copies some resources, like job's JAR file, configuration file, and computed input splits, to the JobTracker's file system

Task assignment

- JobTracker creates one map task for each computed input split by the user node and assigns the map tasks to the execution slots of the TaskTrackers
- It also creates reduce tasks and assigns them to the TaskTrackers. The number of reduce tasks is predetermined by the user

Running a Job in Hadoop

Task Execution

- control flow to execute a task (map/reduce) starts inside the TaskTracker by copying the job JAR file to its file system.
- Instructions inside the job JAR file are executed after launching a Java Virtual Machine (JVM) to run its map or reduce task

Task running check

- It is performed by receiving periodic heartbeat messages to the JobTracker from the TaskTrackers.
- It notifies the JobTracker that the sending TaskTracker is alive, and whether the sending TaskTracker is ready to run a new task