MapReduce Programming Model

MapReduce is a programming paradigm for processing large datasets in a distributed environment. It consists of three main phases, illustrated using an example with a text file (Input.txt) containing fruit names (e.g., "apple, banana", "banana, orange", etc.):

1. **Map Phase**:
   - Input data is split into key-value pairs.
   - For each fruit in the input, a mapper outputs a pair like (fruit, 1) (e.g., apple, 1, banana, 1).
   - Example: "apple, banana" produces (apple, 1), (banana, 1).
2. **Shuffle & Sort**:
   - The system groups and sorts the mapper outputs by key.
   - All pairs with the same key (e.g., all apple entries) are collected together.
3. **Reduce Phase**:
   - A reducer aggregates the values for each key.
   - Example: For apple, it sums the 1s to produce (apple, 3); for banana, (banana, 5); for orange, (orange, 6).
   - Final output: (apple, 3), (banana, 5), (orange, 6).

Anatomy of File Operations

- **File Read**: Refers to how data is accessed from HDFS (details in *Hadoop: The Definitive Guide*).
- **File Write**: Refers to how data is written to HDFS (details in *Hadoop: The Definitive Guide*).

Job Completion and Failure Handling

- **Job Completion**: A job completes when all tasks (map and reduce) are successfully executed.
- **Failure Scenarios**:
  - **Task Failure**: Individual task failure can be retried.
  - **Task Tracker Failure**: Failure of a node running tasks may require reassignment.
  - **Job Tracker Failure**: Critical failure affecting job coordination (relevant in Hadoop 1.x).
  - **NameNode Failure**: Loss of metadata management, a single point of failure in Hadoop 1.x.

HDFS Components (Hadoop 2.x)

Hadoop 2.x introduces improvements over 1.x, addressing scalability and reliability:

1. **NameNode (High Availability)**:
   - Unlike Hadoop 1.x, where the NameNode was a single point of failure, 2.x supports high availability with active and standby NameNodes to ensure continuous operation.
2. **YARN (Yet Another Resource Negotiator)**:
   - Replaces the Job Tracker and Task Tracker model of Hadoop 1.x.
   - YARN is a resource management system that manages cluster resources and schedules tasks.
   - Key components:
     - **Resource Manager**: Oversees resource allocation and job scheduling across the cluster.
     - **Node Manager**: Manages resources and task execution on individual nodes.
     - **Application Master**: Coordinates the execution of a specific application (e.g., a MapReduce job).
   - YARN improves scalability, supports MapReduce, and enables other distributed programming paradigms.

Anatomy of Job Run in YARN

- A job in YARN involves:
  - Submission to the Resource Manager.
  - Allocation of resources by the Resource Manager.
  - Creation of an Application Master to manage the job.
  - Execution of tasks by Node Managers under the Application Master's coordination.
- **Job Completion in YARN**: The job completes when all tasks are executed, with YARN handling resource allocation and fault tolerance dynamically.