Task Execution

Task Execution Environment

- ✓ The MapReduce framework executes mapper and reducer tasks in a **Separate JVM Process.**

- ✓ This process inherits the environment of the parent TaskTracker process.

- ✓ The user can specify additional options to the child JVM via the

Mapreduce.Map.Java.Opts,

Mapreduce.Reduce.Java.Opts

configuration parameters.

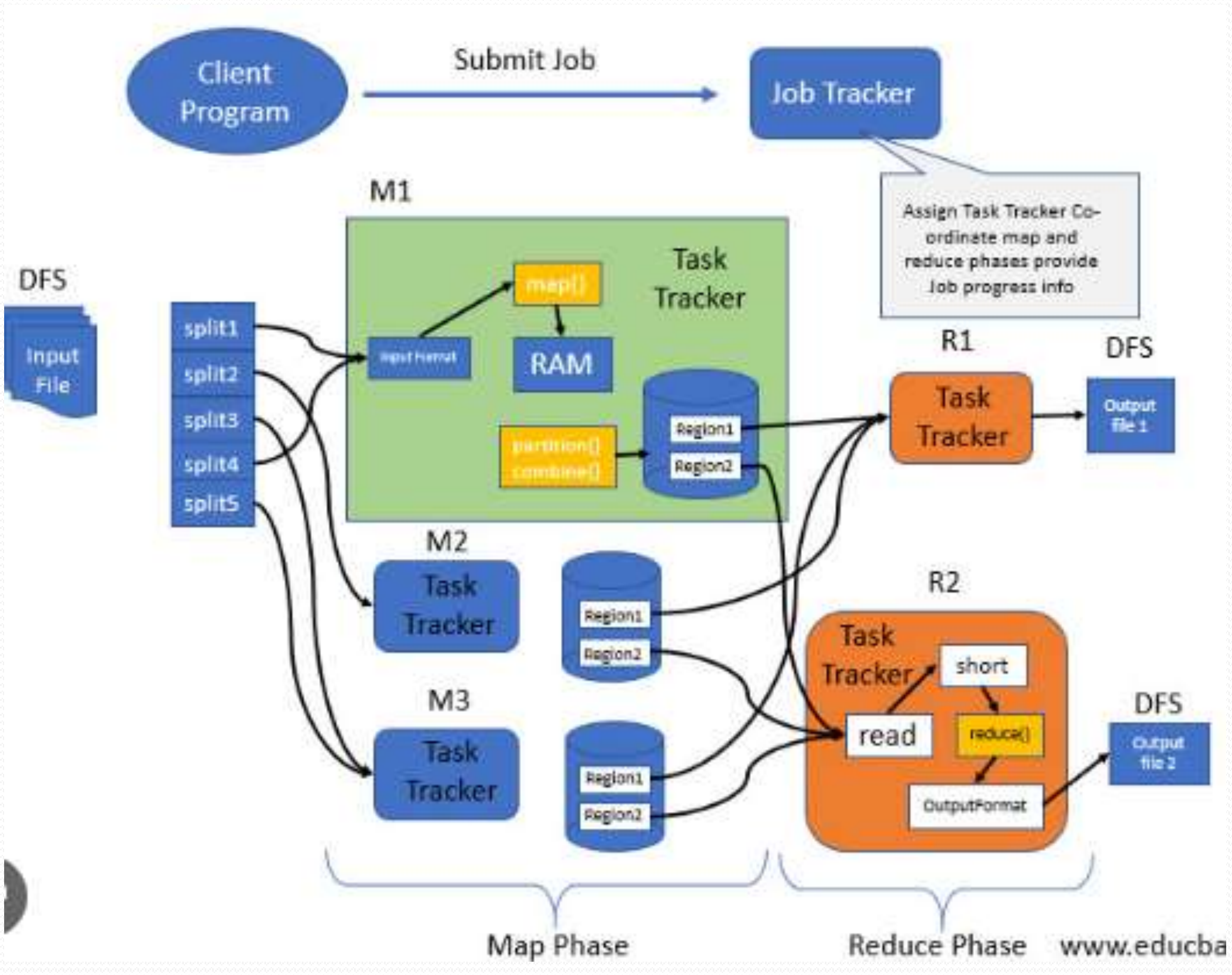
The two essential components of the MapReduce framework are

1. **JobTracker**
2. **TaskTracker**

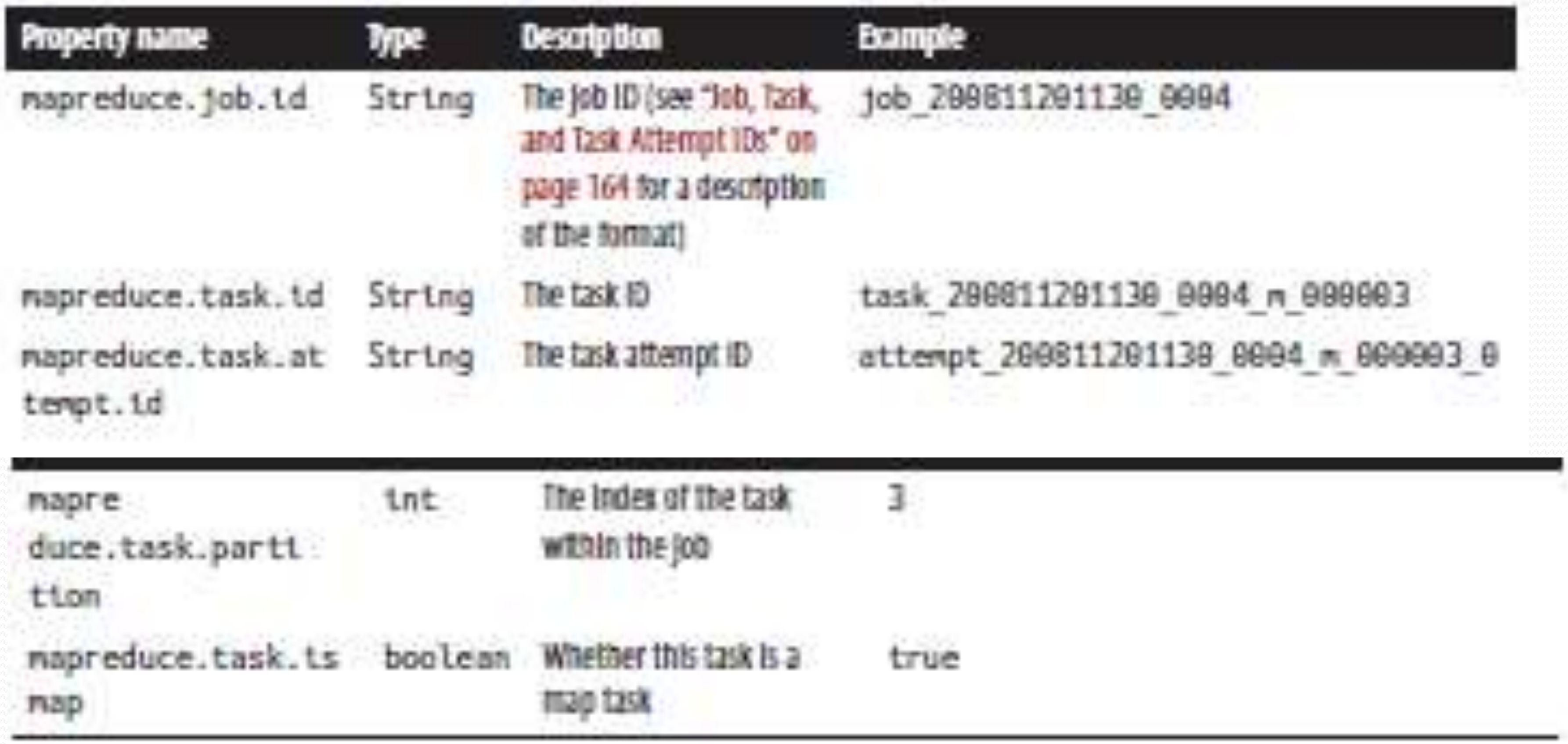
- **Job Tracker** is responsible for **Scheduling And Monitoring** MapReduce jobs.

- **Task Tracker** is responsible for executing **Individual Mapper And Reducer Tasks**.

MapReduce Architecture



Task Environment Properties



Property name	Type	Description	Example
mapreduce.job.id	String	The job ID (see "Job, Task, and Task Attempt IDs" on page 164 for a description of the format)	job_200811201130_0004
mapreduce.task.id	String	The task ID	task_200811201130_0004_m_000003
mapreduce.task.attempt.id	String	The task attempt ID	attempt_200811201130_0004_m_000003_0
mapreduce.task.partition	int	The index of the task within the job	3
mapreduce.task.ismap	boolean	Whether this task is a map task	true

Speculative Execution – Killing The Slowest Copies

- ✓ Speculative execution is a technique used to improve the performance of MapReduce jobs by **Running Multiple Copies Of The Same Task And Killing The Slowest Copies**.

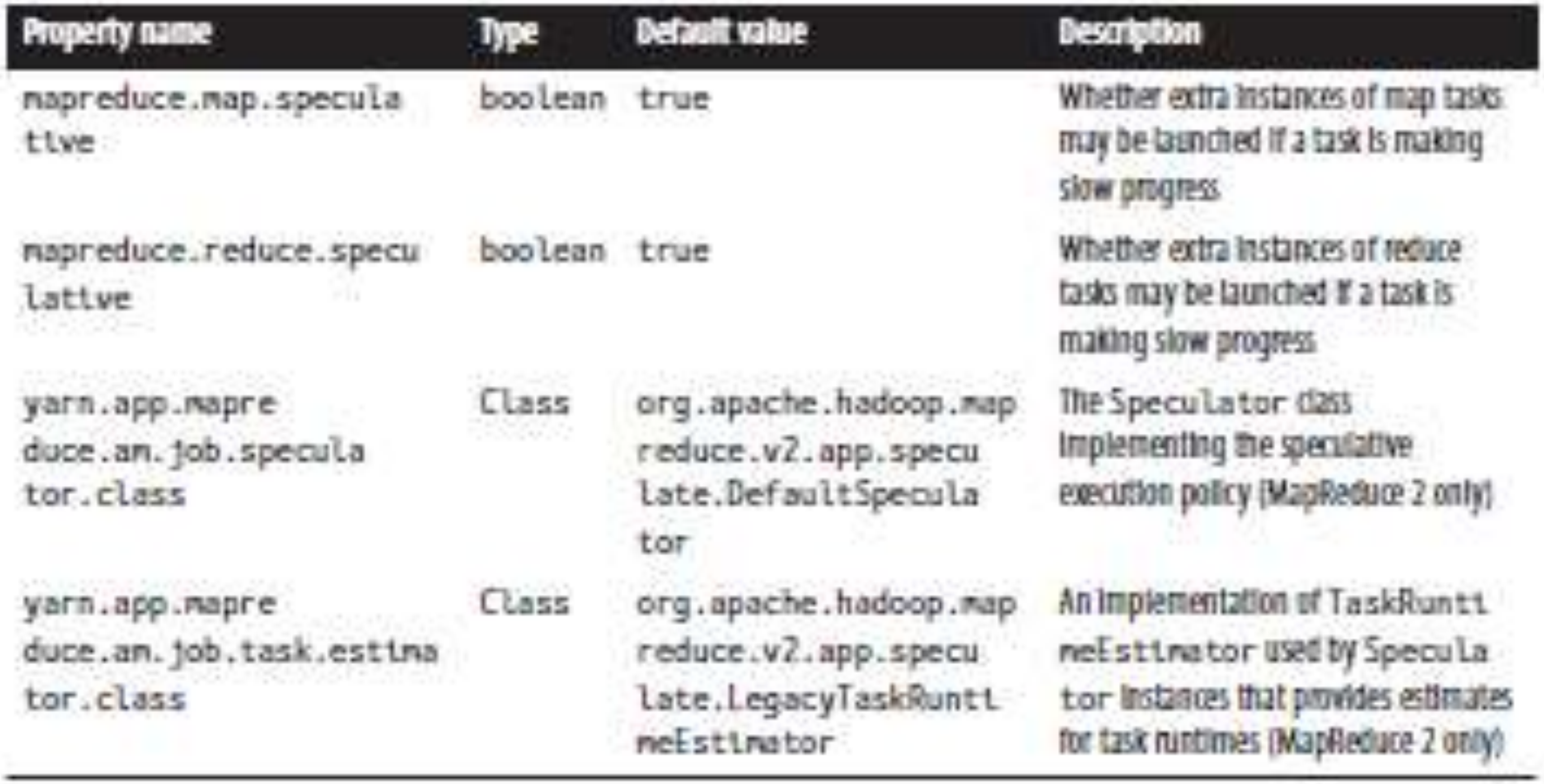
- ✓ The MapReduce framework determines which tasks to speculate on based on their progress and the overall state of the job.

- ✓ Speculative execution can be **Enabled Or Disabled** by setting the **mapreduce.map.speculative,**

mapreduce.reduce.speculative

configuration parameters.

Speculative Execution Properties



Property name	Type	Default value	Description
mapreduce.map.speculative	boolean	true	Whether extra instances of map tasks may be launched if a task is making slow progress
mapreduce.reduce.speculative	boolean	true	Whether extra instances of reduce tasks may be launched if a task is making slow progress
yarn.app.mapreduce.am.job.speculator.class	Class	org.apache.hadoop.mapreduce.v2.app.speculate.DefaultSpeculator	The Speculator class implementing the speculative execution policy (MapReduce 2 only)
yarn.app.mapreduce.am.job.task.estimator.class	Class	org.apache.hadoop.mapreduce.v2.app.speculate.LegacyTaskRuntimeEstimator	An implementation of TaskRuntimeEstimator used by Speculator instances that provides estimates for task runtimes (MapReduce 2 only)

Output Committers – Output is Durable& Consistent

- The output committer is responsible for writing the **output** of a MapReduce job to the Hadoop Distributed File System (**HDFS**).

- The MapReduce framework provides a number of built-in output committers, but the user can also implement their own custom output committer.

- The output committer is responsible for ensuring that the output of the job is durable and consistent (output files are **Not Corrupted And That They Are Properly Synchronized** with the HDFS metadata)

Task Execution Flow

The following is a simplified overview of the task execution flow in MapReduce:

- ✓ The **JobTracker** assigns a mapper or reducer task to a **TaskTracker**.
- ✓ The **TaskTracker** starts a new JVM process to execute the task.
- ✓ The **Task Execution Environment** is initialized with
  - **The Input And Output Splits**
  - **The Mapper Or Reducer Class**
  - **The Configuration Parameters**
  - **The Reporter Object**

- ✓ The task is executed.
- ✓ The **Output Committer** is used to write the output of the task to HDFS.
- ✓ The task execution environment is cleaned up.

If **Speculative Execution** is enabled, the JobTracker may start multiple copies of the same task and kill the slowest copies.