1. How can I avoid excessive buffering when writing scalable Hadoop algorithms?
2. Emit Early and Often: Instead of accumulating intermediate results in memory (like a List or Map), output key-value pairs as soon as possible.
3. Avoid Building Large In-Memory Structures.
4. Use Writable Reuse.
5. Use Combiner to Reduce Buffer Load.
6. Why is preserving state in the Mapper (like using a hash table) faster than using actual combiners in MapReduce?
7. Avoids Object Serialization & Disk I/O

Combiner: Mapper emits intermediate (key, value) pairs → written to memory buffer → sorted → possibly spilled to disk → combiner merges them.

In-Mapper Aggregation: No (key, value) emitted until the end. It aggregates in memory only.

1. Runs Always, Not Opportunistically

Combiner is optional and not guaranteed to run.

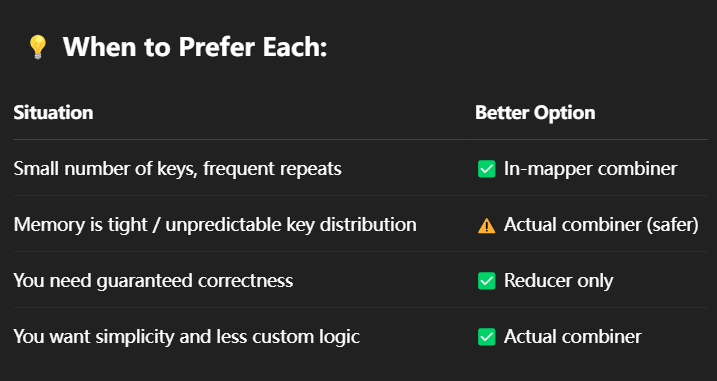
Hadoop may choose not to run a combiner, or only run it partially (e.g., once per spill).

1. Reduces Key Count Early

Combiner runs after spilling, often on many repeated keys.

In-mapper aggregation collapses identical keys as they appear.

1. So, in most cases, are in-mapper combiners always better than actual combiners?



1. What exactly does the shuffling phase in MapReduce do?
2. Partitioning

Assigns each key to a reducer using a partitioner (e.g., hash(key) % numReducers).

This forms the key-to-reducer mapping.

1. Transfer

Moves intermediate (key, value) pairs from mappers to their assigned reducers over the network.

1. Merging + Sorting
2. Grouping for Reduce Function

The framework ensures that the reducer's reduce(K, Iterable<V>).

1. Buffering values in memory is not a good idea.
2. Memory limitation:

You don't know how many values will arrive for each key. If too many values accumulate, it can overflow memory, leading to crashes or out-of-memory errors.

1. Scalability issue:

Works for small datasets, but not big data. If your reducer sees thousands or millions of values for one key, buffering them all is dangerous.

1. Garbage Collection overhead:

The memory pressure can lead to frequent GC, hurting performance even if it doesn't crash.