Hadoop Pluggable Sort

Table of contents

1. Introduction

The pluggable sort capability allows replacing the built in sort logic with alternate implementations. Example use cases for this are replacing the sort logic with custom algorithms that enable Hash aggregation and Limit-N query.

IMPORTANT: The pluggable sort capability is experimental and unstable. This means the provided APIs may change and break compatibility in future versions of Hadoop.

2. Implementing a Custom Sort

A custom sort implementation requires a **org.apache.hadoop.mapred.MapOutputCollector** implementation class running in the Mapper tasks and (optionally, depending on the sort implementation) a **org.apache.hadoop.mapred.ShuffleConsumerPlugin** implementation class running in the Reducer tasks.

The default implementations provided by Hadoop can be used as references:

- **org.apache.hadoop.mapred.MapTask$MapOutputBuffer**
- **org.apache.hadoop.mapreduce.task.reduce.Shuffle**

3. Configuration

All the pluggable components run in the job tasks. This means, they can be configured on per job basis.

Job Configuration Properties

- **mapreduce.job.reduce.shuffle.consumer.plugin.class**. Default:**org.apache.hadoop.mapreduce.task.reduce.Shuffle**. The **ShuffleConsumerPlugin** implementation to use
- **mapreduce.job.map.output.collector.class**. Default:**org.apache.hadoop.mapred.MapTask$MapOutputBuffer**. The **MapOutputCollector** implementation to use

These properties can also be set in the **mapred-site.xml** to change the default values for all jobs.