Memory Management

Users/admins can also specify the maximum virtual memory of the launched child-task, and any sub-process it launches recursively, using `mapred.{map|reduce}.child.ulimit`. Note that the value set here is a per process limit. The value for `mapred.{map|reduce}.child.ulimit` should be specified in kilo bytes (KB). And also the value must be greater than or equal to the -Xmx passed to JavaVM, else the VM might not start.

Note: `mapred.{map|reduce}.child.java.opts` are used only for configuring the launched child tasks from task tracker. Configuring the memory options for daemons is documented in Configuring the Environment of the Hadoop Daemons.

The memory available to some parts of the framework is also configurable. In map and reduce tasks, performance may be influenced by adjusting parameters influencing the concurrency of operations and the frequency with which data will hit disk. Monitoring the filesystem counters for a job- particularly relative to byte counts from the map and into the reduce- is invaluable to the tuning of these parameters.

Users can choose to override default limits of Virtual Memory and RAM enforced by the task tracker, if memory management is enabled. Users can set the following parameter per job:

Name	Type	
`mapred.task.maxvmem`	int	A number, in bytes, that represents the maximum Virtual Memory task-limit fo number.
mapred.task.maxpmem	int	A number, in bytes, that represents the maximum RAM task-limit for each task tasks on a node based on RAM needs.

Map Parameters

A record emitted from a map will be serialized into a buffer and metadata will be stored into accounting buffers. As described in the following options, when either the serialization buffer or the metadata exceed a threshold, the contents of the buffers will be sorted and written to disk in the background while the map continues to output records. If either buffer fills completely while the spill is in progress, the map thread will block. When the map is finished, any remaining records are written to disk and all on-disk segments are merged into a single file. Minimizing the number of spills to disk can decrease map time, but a larger buffer also decreases the memory available to the mapper.

Name	Type	
io.sort.mb	int	The cumulative size of the serialization and accounting buffers storing records em
io.sort.record.percent	float	The ratio of serialization to accounting space can be adjusted. Each serialized reco This percentage of space allocated from `io.sort.mb` affects the probability of a s Clearly, for a map outputting small records, a higher value than the default will lik
io.sort.spill.percent	float	This is the threshold for the accounting and serialization buffers. When this percen Let`io.sort.record.percent` be *r*, `io.sort.mb` be *x*, and this value be *q*. The `2^16`. Note that a higher value may decrease the number of- or even eliminate- r average map times are usually obtained by accurately estimating the size of the r

Other notes

- If either spill threshold is exceeded while a spill is in progress, collection will continue until the spill is finished. For example, if `io.sort.buffer.spill.percent` is set to 0.33, and the remainder of the buffer is filled while the spill runs, the next spill will include all the collected records, or 0.66 of the buffer, and will not generate additional spills. In other words, the thresholds are defining triggers, not blocking.
- A record larger than the serialization buffer will first trigger a spill, then be spilled to a separate file. It is undefined whether or not this record will first pass through the combiner.