Capacity Scheduler

Table of contents

1 Purpose

This document describes the Capacity Scheduler, a pluggable MapReduce scheduler for Hadoop which provides a way to share large clusters.

2 Features

The Capacity Scheduler supports the following features:

- Multiple queues, possibly hierarchical/recursive, where a job is submitted to a queue.
- Queues are allocated a fraction of the capacity of the grid in the sense that a certain capacity of resources will be at their disposal. All jobs submitted to a queue will have access to the capacity allocated to the queue.
- Free resources can be allocated to any queue beyond it's capacity. When there is demand for these resources from queues running below capacity at a future point in time, as tasks scheduled on these resources complete, they will be assigned to jobs on queues running below the capacity.
- Queues optionally support job priorities (disabled by default).
- Within a queue, jobs with higher priority will have access to the queue's resources before jobs with lower priority. However, once a job is running, it will not be preempted for a higher priority job, though new tasks from the higher priority job will be preferentially scheduled.
- In order to prevent one or more users from monopolizing its resources, each queue enforces a limit on the percentage of resources allocated to a user at any given time, if there is competition for them.
- Queues can use idle resources of other queues. In order to prevent monopolizing of resources by particular queues, each queue can be set a cap on the maximum number of resources it can expand to in the presence of idle resources in other queues of the cluster.
- Support for memory-intensive jobs, wherein a job can optionally specify higher memory-requirements than the default, and the tasks of the job will only be run on TaskTrackers that have enough memory to spare.
- Support for refreshing/reloading some of the queue-properties without restarting the JobTracker, taking advantage of the queue-refresh feature in the framework.

3 Picking a Task to Run

Note that many of these steps can be, and will be, enhanced over time to provide better algorithms.

Whenever a TaskTracker is free, the Capacity Scheduler picks a queue which has most free space (whose ratio of # of running slots to capacity is the lowest).

Once a queue is selected, the Scheduler picks a job in the queue. Jobs are sorted based on when they're submitted and their priorities (if the queue supports priorities). Jobs are considered in order, and a job is selected if its user is within the user-quota for the queue, i.e., the user is not already using queue resources above his/her limit. The Scheduler also makes sure that there is enough free memory in the TaskTracker to tun the job's task, in case the job has special memory requirements.

Once a job is selected, the Scheduler picks a task to run. This logic to pick a task remains unchanged from earlier versions.

3.1 Scheduling Tasks Considering Memory Requirements

The Capacity Scheduler supports scheduling of tasks on a TaskTracker based on a job's virtual memory requirements and the availability of enough virtual memory on the TaskTracker node. By doing so, it simplifies the virtual memory monitoring function on the TaskTracker node, described in the section on Monitoring Task Memory Usage in the Cluster Setup guide. Refer to that section for more details on how memory for MapReduce tasks is handled.

Virtual memory based task scheduling uses the same parameters as the memory monitoring function of the TaskTracker, and is enabled along with virtual memory monitoring. When enabled, the scheduler ensures that a task is scheduled on a TaskTracker only when the virtual memory required by the map or reduce task can be assured by the TaskTracker. That is, the task is scheduled only if the following constraint is satisfied:

```
Job's mapreduce.{map|reduce}.memory.mb of the job <= total
virtual memory for all map or reduce tasks on the TaskTracker
- total virtual memory required for all running map or reduce
tasks on the TaskTracker
```

When a task at the front of the scheduler's queue cannot be scheduled on a TaskTracker due to insufficient memory, the scheduler creates a virtual *reservation* for this task. This can continue for all pending tasks on a job, subject to other capacity constraints. Once all tasks are either scheduled or have reservations, the scheduler will proceed to schedule other jobs's tasks that are not necessarily at the front of the queue, but meet memory constraints of the TaskTracker. By following this reservation procedure of reserving just enough TaskTrackers, the scheduler balances between not starving jobs with high memory requirements and under-utilizing cluster resources.

Tasks of jobs that require more virtual memory than the per slot `mapreduce.cluster.{map|reduce}memory.mb` value, are treated as occupying more than one slot, and account for a corresponding increased capacity usage for their queue. The number of slots they occupy is determined as:

`Number of slots for a task = mapreduce.{map|reduce}.memory.mb / mapreduce.cluster.{map|reduce}memory.mb`
However, special tasks run by the framework like setup and cleanup tasks do not count for more than 1 slot, irrespective of their job's memory requirements.

4 Installation

The Capacity Scheduler is available as a JAR file in the Hadoop tarball under the *contrib/capacity-scheduler* directory. The name of the JAR file would be on the lines of hadoop-*-capacity-scheduler.jar.

You can also build the Scheduler from source by executing *ant package*, in which case it would be available under *build/contrib/capacity-scheduler*.

To run the Capacity Scheduler in your Hadoop installation, you need to put it on the *CLASSPATH*. The easiest way is to copy the `hadoop-*-capacity-scheduler.jar` from to `HADOOP_HOME/lib`. Alternatively, you can modify *HADOOP_CLASSPATH* to include this jar, in `conf/hadoop-env.sh`.

5 Configuration

5.1 Using the Capacity Scheduler

To make the Hadoop framework use the Capacity Scheduler, set up the following property in the site configuration:

Name	Value
mapreduce.jobtracker.taskscheduler	org.apache.hadoop.mapred.CapacityTaskScheduler

5.2 Setting Up Queues

You can define multiple, possibly hierarchical queues to which users can submit jobs with the Capacity Scheduler. To define queues, various properties should be set in two configuration files - mapred-queues.xml and conf/capacity-scheduler.xml .

conf/capacity-scheduler.xml can be used to configure (1) job-initialization-poller related properties and (2) the default values for various properties in the queues

conf/mapred-queues.xml contains the actual queue configuration including (1) framework specific properties like ACLs for controlling which users or groups have access to the queues and state of the queues and (2) the scheduler specific properties for each queue. If any of these scheduler specific properties are missing and not configured for a queue, then the properties in *conf/capacity-scheduler.xml* are used to set default values. More details about the properties that can be configured, and their semantics is mentioned below. Also, a default

template for mapred-queues.xml tailored for using with Capacity-scheduler can be found here.

5.3 Configuring Properties for Queues

The Capacity Scheduler can be configured with several properties for each queue that control the behavior of the Scheduler. As described above, this scheduler specific configuration has to be in the *conf/mapred-queues.xml* along with the rest of the framework specific configuration. By default, the configuration is set up for one queue, named *default*.

To specify a property for a specific queue that is defined in the mapred-queues.xml, you should set the corresponding property in a <property> tag explained here.

The properties defined for queues and their descriptions are listed in the table below:

Name	Refresh-able?	Applicable to?	Description
capacity	Yes	Container queues as well as leaf queues	For a root-level container queue, this is the percentage of the number of slots in the cluster that will be available for all its immediate children together. For a root-level leaf-queue, this is the percentage of the number of slots in the cluster that will be available for all its jobs. For a non-root level container queue, this is the percentage of the number of slots in its parent queue that will be available for all its children together. For a non-root-level leaf queue, this is the percentage of the number of slots in its parent queue that will be available for jobs in this queue. The sum of capacities for all children of a container queue should be less than or equal 100. The sum of capacities of all the root-

Name	Refresh-able?	Applicable to?	Description
			level queues should be less than or equal to 100.
maximum-capacity	Yes	Container queues as well as leaf queues	A limit in percentage beyond which a non-root-level queue cannot use the capacity of its parent queue; for a root-level queue, this is the limit in percentage beyond which it cannot use the cluster-capacity. This property provides a means to limit how much excess capacity a queue can use. It can be used to prevent queues with long running jobs from occupying more than a certain percentage of the parent-queue or the cluster, which, in the absence of pre-emption, can lead to capacity guarantees of other queues getting affected. The maximum-capacity of a queue can only be greater than or equal to its capacity. By default, there is no limit for a queue. For a non-root-level queue this means it can occupy till the maximum-capacity of its parent, for a root-level queue, it means that it can occupy the whole cluster. A value of 100 implies that a queue can use the complete capacity of its parent, or the complete cluster-capacity in case of root-level-queues.

Name	Refresh-able?	Applicable to?	Description
supports-priority	No	Leaf queues only	If true, priorities of jobs will be taken into account in scheduling decisions.
minimum-user-limit-percent	Yes	Leaf queues only	Each queue enforces a limit on the percentage of resources allocated to a user at any given time, if there is competition for them. This user limit can vary between a minimum and maximum value. The former depends on the number of users who have submitted jobs, and the latter is set to this property value. For example, suppose the value of this property is 25. If two users have submitted jobs to a queue, no single user can use more than 50% of the queue resources. If a third user submits a job, no single user can use more than 33% of the queue resources. With 4 or more users, no user can use more than 25% of the queue's resources. A value of 100 implies no user limits are imposed.
maximum-initialized-jobs-per-user	Yes	Leaf queues only	Maximum number of jobs which are allowed to be pre-initialized for a particular user in the queue. Once a job is scheduled, i.e. it starts running, then that job is not considered while scheduler computes the maximum job a user is allowed to initialize.

See this configuration file for a default configuration of queues in capacity-scheduler.

5.4 Job Initialization Parameters

Capacity scheduler lazily initializes the jobs before they are scheduled, for reducing the memory footprint on jobtracker. Following are the parameters, by which you can control the laziness of the job initialization. The following parameters can be configured in capacity-scheduler.xml:

Name	Description
mapred.capacity-scheduler.init-poll-interval	Amount of time in miliseconds which is used to poll the scheduler job queue to look for jobs to be initialized.
mapred.capacity-scheduler.init-worker-threads	Number of worker threads which would be used by Initialization poller to initialize jobs in a set of queue. If number mentioned in property is equal to number of job queues then a thread is assigned jobs from one queue. If the number configured is lesser than number of queues, then a thread can get jobs from more than one queue which it initializes in a round robin fashion. If the number configured is greater than number of queues, then number of threads spawned would be equal to number of job queues.

5.5 Reviewing the Configuration of the Capacity Scheduler

Once the installation and configuration is completed, you can review it after starting the MapReduce cluster from the admin UI.

- Start the MapReduce cluster as usual.
- Open the JobTracker web UI.
- The queues you have configured should be listed under the *Scheduling Information* section of the page.
- The properties for the queues should be visible in the *Scheduling Information* column against each queue.