Hadoop 公平调度器指南

翻译自 Hadoop 包中的 docs/capacity_scheduler.html，版本 Cloudera Hadoop 0.20.1+152

Spork
gmspork@gmail.com

目　录

目的.....1
引言.....1
安装.....1
配置.....2
    mapred-site.xml 中的调度器参数.....2
        基本参数.....2
        高级参数.....3
    配额文件格式.....4
管理.....5
实现.....5

目的

本文档描述了公平调度器（Fair Scheduler），这是一个用于 Hadoop 的插件式的 Map/Reduce 调度器，它提供了一种共享大规模集群的方法。

引言

公平调度是一种赋予作业（job）资源的方法，它的目的是让所有的作业随着时间的推移，都能平均的获取等同的共享资源。当单独一个作业在运行时，它将使用整个集群。当有其它作业被提交上来时，系统会将任务（task）空闲时间片（slot）赋给这些新的作业，以使得每一个作业都大概获取到等量的 CPU 时间。与 Hadoop 默认调度器维护一个作业队列不同，这个特性让小作业在合理的时间内完成的同时又不“饿”到消耗较长时间的大作业。它也是一个在多用户间共享集群的简单方法。公平共享可以和作业优先权搭配使用——优先权像权重一样用作为决定每个作业所能获取的整体计算时间的比例。

公平调度器按资源池（pool）来组织作业，并把资源公平的分到这些资源池里。默认情况下，每一个用户拥有一个独立的资源池，以使每个用户都能获得一份等同的集群资源而不管他们提交了多少作业。按用户的 Unix 群组或作业配置（jobconf）属性来设置作业的资源池也是可以的。在每一个资源池内，会使用公平共享（fair sharing）的方法在运行作业之间共享容量（capacity）。你也可以给予资源池相应的权重，以不按比例的方式共享集群。

除了提供公平共享方法外，公平调度器允许赋给资源池保证（guaranteed）最小共享资源，这个用在确保特定用户、群组或生产应用程序总能获取到足够的资源时是很有用的。当一个资源池包含作业时，它至少能获取到它的最小共享资源，但是当资源池不完全需要它所拥有的保证共享资源时，额外的部分会在其它资源池间进行切分。

在常规操作中，当提交了一个新作业时，公平调度器会等待已运行作业中的任务完成以释放时间片给新的作业。但，公平调度器也支持在可配置的超时时间后对运行中的作业进行抢占。如果新的作业在一定时间内还获取不到最小的共享资源，这个作业被允许去终结已运行作业中的任务以获取运行所需要的资源。因此抢占可以用来保证“生产”作业在指定时间内运行的同时也让 Hadoop 集群能被实验或研究作业使用。另外，作业的资源在可配置的超时时间（一般设置大于最小共享资源超时时间）内拥有不到其公平共享资源（fair share）的一半的时候也允许对任务进行抢占。在选择需要结束的任务时，公平调度器会在所有作业中选择那些最近运行起来的任务，以最小化被浪费的计算。抢占不会导致被抢占的作业失败，因为 Hadoop 作业能容忍丢失任务，这只是会让它们的运行时间更长。

最后，公平调度器还可以限制每用户和每资源池的并发运行作业数量。当一个用户必须一次性提交数百个作业时，或当大量作业并发执行时，用来确保中间数据不会塞满集群上的磁盘空间，这是很有用的。设置作业限制会使超出限制的作业被列入调度器的队列中进行等待，直到一些用户/资源池的早期作业运行完毕。系统会根据作业优先权和提交时间的排列来运行每个用户/资源池中的作业。

安装

要让公平调度器能在你的 Hadoop 中运行，你需要把它放到 CLASSPATH 中。最简单的方法就是把 *hadoop-*-fairscheduler.jar* 从 *HADOOP_HOME/build/contrib/fairscheduler* 拷贝到 *HADOOP_HOME/lib* 。你也可以修改 *HADOOP_CONF_DIR/hadoop-env.sh* 中的 *HADOOP_CLASSPATH*，加入公平调度器的 jar 包。

你还需要在 Hadoop 的配置文件 *HADOOP_CONF_DIR/mapred-site.xml* 中设置下列属性让 Hadoop 使用公平调度器：

```
<property>
  <name>mapred.jobtracker.taskScheduler</name>
  <value>org.apache.hadoop.mapred.FairScheduler</value>
</property>
```

在重启集群后，你可以通过 JobTracker 的 web 用户界面中的 *http://<jobtracker URL>/scheduler* 检查公平调度器是否正在运行。你应该能看到一个"job scheduler administration"页面。我们将在管理章节讲述这个页面。

如果你想从源码中编译公平调度器，请在你的 *HADOOP_HOME* 目录中运行 *ant package*。这个操作将会构建 *build/contrib/fair-scheduler/hadoop-*-fairscheduler.jar*。

配置

公平调度器有两处配置文件——算法参数在 *mapred-site.xml* 中设置，还有一个单独的称为 *配额文件（allocation file）*的 XML 文件，可以用来配置资源池、最小共享资源、运行作业限制和抢占超时时间。配额文件在运行时会定期被重新加载，这可以让你修改资源池的设置而不用重启 Hadoop 集群。

对于仅需要在用户间获取等同共享的最小化安装，你就不需要去配置配额文件了。如果对配额文件进行了配置，你需要设置 *mapred-site.xml* 中的 *mapred.fairscheduler.allocation.file*（下面描述）参数告诉调度器怎么去找到配额文件。

mapred-site.xml 中的调度器参数

可以在 *mapred-site.xml* 中设置下面的参数来影响公平调度器的行为：

基本参数

属性名	描述
mapred.fairscheduler.allocation.file	指定一个 XML 文件的绝对路径，该文件包含了每个资源池的最小共享资源、每资源池和每用户的并发运行作业数和抢占超时时间。如果没有设置这个属性，这些特性将不会被使用。配额文件格式在稍后描述。
mapred.fairscheduler.preemption	是否启用抢占的布尔值属性。默认是 false。
mapred.fairscheduler.poolnameproperty	指定用哪个作业配置属性来决定作业的归属资源池。字符串格式，默认：*user.name*（即是每个用户一个资源池）。另一个有用的值是 *group.name*，即每个 Unix 群组一个资源池。一个常用的设定是使用非标准的属性如 *pool.name* 作为资源池的名字属性，然后通过添加下面的设定来使 *user.name* 成为默认： <property> <name>pool.name</name> <value>${user.name}</value> </property> 这样你就可以对某些作业显式的通过作业配置属

	性来指定资源池的名字（比如，在有默认用户资源池的情况下，传递 *-Dpool.name=<name>* 到 *bin/hadoop jar*）。

高级参数

属性名	描述
mapred.fairscheduler.sizebasedweight	在计算作业的公平共享权重时考虑作业大小。默认情况下，权重只基于作业的优先权。设置这个标志为 true 会使权重也考虑作业大小（所需任务数），但不是线性的（权重与所需任务数的对数成比例）。这个设定让较大作业在获取更大的公平共享资源的同时也能提供足够的共享资源给小作业，让它们能迅速的完成。布尔值，默认是 false。
mapred.fairscheduler.preemption.only.log	这个标志会使调度器在碰到抢占计算时仅简单的记录下它什么时候想抢占一个任务，而不会真正的抢占任务。布尔值，默认是 false。这个属性用在启用抢占之前做一个抢占的“dry run”是很有用的，以确保你没把超时时间设置的过于具有侵略性。你会在 Jobtracker 的输出日志（*HADOOP_LOG_DIR/hadoop-jobtracker-*.log*）看到抢占日志信息。信息跟下面的相似： Should preempt 2 tasks for job_20090101337_0001: tasksDueToMinShare = 2, tasksDueToFairShare = 0
mapred.fairscheduler.update.interval	公平共享资源计算更新间隔时间。默认的 500 毫秒适用于小于 500 个节点的集群，但较大的值可以减少更大集群的 Jobtracker 的负载。整数值，单位是毫秒，默认是 500。
mapred.fairscheduler.preemption.interval	检查任务抢占的间隔时间。默认的 15 秒适用于超时时间在分钟数量级上的。不推荐超时时间过小于这个数值，但是如果你已经设置了这样的超时时间，你可以使用这个值来做更多的抢占计算。然而小于 5 秒的值就太小了，因为它小于心跳的间隔时间了。整数值，单位是毫秒，默认是 15000。
mapred.fairscheduler.weightadjuster	一个扩展点，让你指定一个类去调整运行中作业的权重。这个类应当实现 *WeightAdjuster* 接口。目前已有一个例子实现——*NewJobWeightBooster*，它会在作业生命周期中的前 5 分钟增加作业的权重，以使小作业能更快速的完成。要使用这个例子实现，设置 weightadjuster 属性为类的全名，org.apache.hadoop.mapred.NewJobWeightBooster。NewJobWeightBooster 本身提供了两个参数用于设定持续时间和增长因子。 ● mapred.newjobweightbooster.factor，新作业权重的增长因子，默认是 3。

	• mapred.newjobweightbooster.duration，增长持续时间，单位是毫秒。默认是 300000，5 分钟。
mapred.fairscheduler.loadmanager	一个扩展点，让你指定一个类去决定一个给定 TaskTracker 上可以运行多少个 map 和 reduce。这个类应当实现 *LoadManager* 接口。默认使用 Hadoop 配置文件中的任务负载，但可以使用这个选项使负载基于如可用内存和 CPU 利用率。
mapred.fairscheduler.taskselector	一个扩展点，让你指定一个类去决定作业内的哪一个任务运行在给定的 tracker 上。这个特性可以用来改变本地化策略（比如，让一些作业在特定机架内）或推测（speculative）执行算法（选择什么时候去执行推测任务）。默认的实现使用 Hadoop 的 JobInProgress 中的默认算法。

配额文件格式

配额文件为每一个资源池配置最小共享资源、运行作业限制、权重和抢占超时时间。*HADOOP_HOME/conf/fair-scheduler.xml.template* 提供了一个示例例子。配额文件可以包含下列类型的元素：

- *pool* 元素，配置各个资源池。它们可能包含下列子元素：
  - *minMaps* 和 *minReduces*，设置资源池最小共享的任务时间片。
  - *maxRunningJobs*，限制从资源池同时运行的作业数量（默认是无限）。
  - *weight*，以非比例的方式与其它资源池共享集群（默认是 1.0）。
  - *minSharePreemptionTimeout*，如果资源池的资源低于它的最小共享资源，在结束其它资源池的任务之前等待的秒数（默认是无限）。
- *user* 元素，可能会包含一个的 *maxRunningJobs* 属性来限制作业。在默认情况下要注意，每一个用户有一个资源池，所以每用户限制不是必须的。
- *poolMaxJobsDefault*，为那些没有指定运行作业限制的资源池设定默认值。
- *userMaxJobsDefault*，为那些没有指定运行作业限制的用户设定默认值。
- *defaultMinSharePreemptionTimeout*，为那些没有指定最小共享资源抢占超时时间的资源池设定默认值。
- *fairSharePreemptionTimeout*，设置作业拥有低于其公平共享资源的一半的抢占超时时间。

Pool 和 user 元素仅在你对资源池/用户设定了非默认值时才是必须的。也就是说，在运行公平调度器前，你不必在配置文件中声明所有的用户及资源池。如果一个用户或资源池没有列在配置文件中，将会使用运行作业限制、抢占超时时间等的默认值。

下面给出一个配额文件示例：

```
<?xml version="1.0"?>
<allocations>
  <pool name="sample_pool">
    <minMaps>5</minMaps>
    <minReduces>5</minReduces>
    <weight>2.0</weight>
  </pool>
```

```
  <user name="sample_user">
    <maxRunningJobs>6</maxRunningJobs>
  </user>
  <userMaxJobsDefault>3</userMaxJobsDefault>
</allocations>
```

这个例子创建了一个资源池 sample_pool，保证最小有 5 个 map 时间片和 5 个 reduce 时间片。资源池还有着 2.0 的权重，意味着它与其它资源池相比，拥有相对 2 倍的集群共享资源（默认是权重是 1）。最后，除了 sample_user 可并行运行 6 个作业外，示例限制了其它每用户的运行作业数为 3。任何未在配额文件中定义的资源池将获得无保证的容量及 1.0 的权重。与此同时，任何没有在配额文件中设置最大运行作业数的资源池或用户将被允许运行无限个作业。

HADOOP_HOME/conf/fair-scheduler.xml.template 是一个更详细的例子文件，还设定了抢占超时时间。

管理

公平调度器通过两种途径提供运行时的管理支持：

1、在运行时修改配额文件中的最小共享资源、运行作业限制、权重和抢占超时时间。调度器在检测到配额文件被修改过后 10-15 秒内会重新加载它。

2、在 JobTracker 的 web 接口 *http://<JobTracker URL>/scheduler* 中查看当前的作业、资源池和公平共享资源。利用这个接口，你可以修改作业的优先权或把作业从一个资源池转移到另外一个，并可以看到在公平共享资源上的效果（这个需要 JavaScript）。

在 web 接口上可以看到下面的各个作业的字段：

- *Submitted* - 作业提交的日期和时间。
- *JobID*，*User*，*Name* - 作业标识，和标准 web 用户接口上的一样。
- *Pool* - 作业当前所属的资源池。选择另外一个值把作业移动到另外一个资源池。
- *Priority* - 当前优先权。选择另外一个值来改变作业的优先权。
- *Maps/Reduces Finished*：已完成的任务数 / 全部任务数。
- *Maps/Reduces Running*：当前正在运行的任务。
- *Map/Reduce Fair Share*：根据公平共享的方法，这个作业在任意时刻应该得到的平均任务时间片。实际的任务时间片会根据作业已有的计算时间上下波动，但一般它会获取到它的公平共享资源的量。

另外，还可以在 *http://<JobTracker URL>/scheduler?advanced* 查看 web 用户接口的“高级”版本。这个会展示更多列：

- *Maps/Reduce Weight*：作业在公平共享计算中的权重。这个依赖于优先权，如果启用了 *sizebasedweight* 和 *NewJobWeightBooster*，那还跟作业的大小及年龄有关。
- *Map/Reduce Deficit*：作业在机器上的调度缺额——根据作业的公平共享资源所应得到资源量，减去它的实际所得。正的缺额意味着作业不久将会被调度，因为它需要达到它的公平共享资源。调度器会首先调度拥有较高缺额的作业。细节请查看本文的实现章节。

实现

实现公平调度分两个方面：计算每个作业的公平共享资源，及当一个任务的时间片可用时选择哪个作业去运行。

选择了运行作业后，调度器会跟踪每一个作业的“缺额”——作业在理想调度器上所应得的计算时间与实际所获得的计算时间的差额。这是一个测量作业的“不公平”待遇的量度标准。每过几百毫秒，调度器就会通过查看各个作业在这个间隔内运行的任务数同它的公平共享资源的差额来更新各个作业的缺额。当有任务时间片可用时，它会被赋给拥有最高缺额的作业。但有一个例外——如果有一个或多个作业都没有达到它们的资源池容量的保证量，我们只在这些“贫穷”的作业间进行选择（再次基于它们的缺额）以保证调度器能尽快的满足资源池的保证量。

公平共享资源是依据各个作业的“权重”，通过在可运行作业之间平分集群容量计算出来的。默认权重是基于作业优先权的，每一级优先权的权重是下一级的 2 倍（比如，VERY_HIGH 的权重是 NORMAL 的 4 倍）。但是，就像在配置章节讲述的一样，权重也可以基于作业的大小和年龄。对于在一个资源池内的作业，公平共享资源还会考虑这个资源池的最小保证量，接着再根据作业的权重在这个资源池内的作业间划分这个容量。

在用户或资源池的运行作业限制没有达到上限的时候，我们做法同标准的 Hadoop 调度器一样，在选择要运行的作业时，首先根据作业的优先权对所有作业进行排序，然后再根据提交时间进行排序。对于上述排序队列中那些超出用户/资源池限制的作业将会被排队并等待空闲时间片，直到它们可以运行。在这段时间内，它们被公平共享计算忽略，不会获得或失去缺额（它们的公平均分量被设为 0）。

抢占是通过定期检查是否有作业的资源低于其最小共享资源或低于其公平共享资源的一半。如果一个作业的资源低于其共享资源的时间足够长，它将被允许去结束其它作业的任务。所选择的任务是所有作业中那些最近运行起来的任务，以最小化被浪费的计算。

最后，公平调度器在可扩展的基本功能上提供了几个扩展点。比如，权重计算可修改为给新作业一个权重增长，实现一个进一步减少交互式作业的响应时间的“最小作业优先”策略。这些扩展点在高级的 mapred-site.xml 属性中有列出。