Hadoop yarn任务调度策略介绍

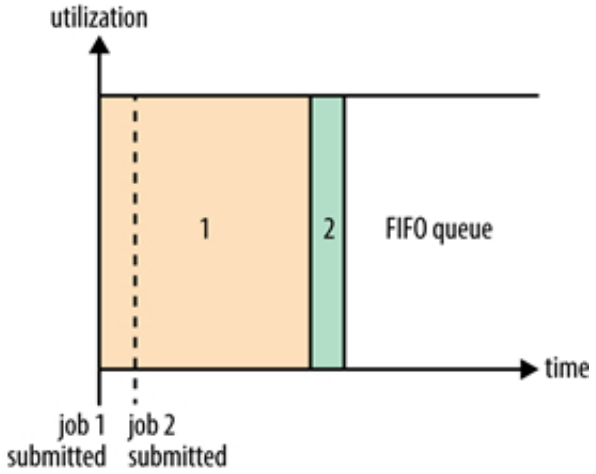
本文将介绍Hadoop YARN提供的三种任务调度策略：FIFO Scheduler，Capacity Scheduler 和 Fair Scheduler。

FIFO Scheduler

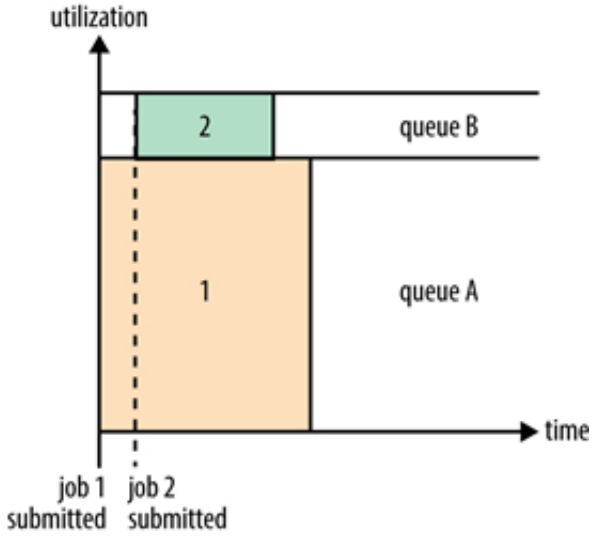
顾名思义，这就是先进先出(first in, first out)调度策略，所有的application将按照提交的顺序来执行，这些 application 都放在一个队列里，只有在执行完一个之后，才会继续执行下一个。

这种调度策略很容易理解，但缺点也很明显。耗时的长任务会导致后提交的任务一直处于等待状态，如果这个集群是多人共享的，显然不太合理。因此 YARN 提供了另外两种调度策略，更加适合共享集群。下图的第一副小图是FIFO Scheduler 执行过程的示意图：

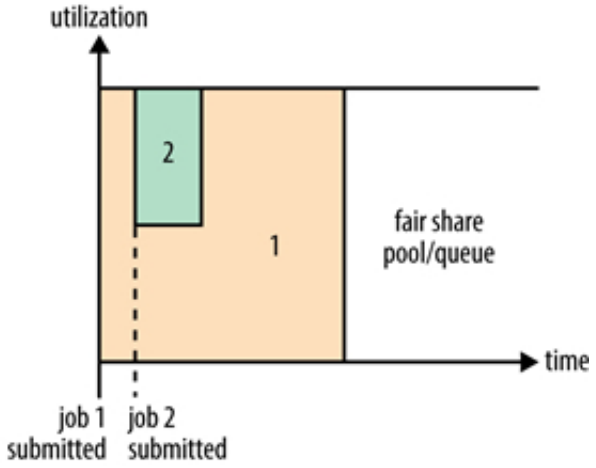
i. FIFO Scheduler



ii. Capacity Scheduler



iii. Fair Scheduler



Capacity Scheduler

前面的FIFO调度策略只适合一个人，而且资源的利用率不是很高。基于这些原因，多人共享的调

度器出现了，Capacity Scheduler就是其中一个。Capacity Scheduler为每个人分配一个队列，每个队列占用的集群资源是固定的，但是可以不同，队列内部还是采用FIFO调度的策略。上图的第二幅小图是Capacity Scheduler 执行过程的示意图。

可以看到，队列 A 和 B 享有独立的资源，但是 A
所占的资源比重更多。如果任务在被执行的时候，集群恰好有空闲资源，比如队列 B
为空，那么调度器就可能分配更多的资源给队列
A，以更好地利用空闲资源。这种处理方式被叫做「queue elasticity」（弹性队列）。

但是弹性队列也有一些副作用，如果此时队列 B 有了新任务，之前被队列 A
占用的资源并不会立即释放，只能等到队列 A
的任务执行完。为了防止某个队列过多占用集群资源，YARN 提供了一个设置可以控制某个队列能够占用的最大资源。但这其实又是跟弹性队列冲突的，因此这里有一个权衡的问题，这个最大值设为多少需要不断试验和尝试。

Capacity Scheduler 的队列是支持层级关系的，即有子队列的概念。对于如下层次结构的队列：

```
root
├── prod
└── dev
    ├── eng
    └── science
```

下面是一个示例配置文件：

```
<?xml version="1.0"?>
<configuration>
    <property>
        <name>yarn.scheduler.capacity.root.queues</name>
        <value>prod,dev</value>
    </property>

    <property>
        <name>yarn.scheduler.capacity.root.dev.queues</name>
        <value>eng,science</value>
    </property>

    <property>
        <name>yarn.scheduler.capacity.root.prod.capacity</name>
        <value>40</value>
    </property>

    <property>
```

```
        <name>yarn.scheduler.capacity.root.dev.capacity</name>
        <value>60</value>
    </property>

    <property>
        <name>yarn.scheduler.capacity.root.dev.maximum-capacity</name>
        <value>75</value>
    </property>

    <property>
        <name>yarn.scheduler.capacity.root.dev.eng.capacity</name>
        <value>50</value>
    </property>

    <property>
        <name>yarn.scheduler.capacity.root.dev.science.capacity</name>
        <value>50</value>
    </property>
</configuration>
```

所有队列的根队列叫做 root，这里一共有两个队列：dev 和 prod，dev
队列之下又有两个子队列：eng 和 science。dev 和 prod 分别占用了 60% 和 40%
的资源比重，同时限制了 dev 队列能够伸缩到的最大资源比重是 75%，换句话说，prod
队列至少能有 25% 的资源分配。eng 和 science 队列各占
50%，但因为没有设置最大值，所以有可能出现某个队列占用整个父队列资源的情况。

除了设置队列层级关系和资源分配比重之外，Capacity Scheduler
还提供了诸如控制每个用户或者任务最大占用资源、同时执行的最大任务数，以及队列的 ACL
等配置，详细请参考官方文档。

队列放置

分配好了队列，要怎么控制任务在指定队列执行呢？如果是 MapReduce 程序，那么可以通过
mapreduce.job.queuename 来设置执行队列，默认情况是在 default
队列执行。注意指定的队列名不需要包含父队列，即不能写成 root.dev.eng，而应该写 eng。

Fair Scheduler

Fair Scheduler 试图为每个任务均匀分配资源，比如当前只有任务 1
在执行，那么它拥有整个集群资源，此时任务 2 被提交，那任务 1 和任务 2
将平分集群资源，以此类推。

当然 Fair Scheduler 也支持队列的概念，上图的第三幅是执行过程的示意图。队列 A 首先执行任务，任务 1 拥有整个集群资源，随后队列 B 增加任务 2，这两个队列按照配置规则获取不同的资源，接着任务2运行完，任务1又拥有整个集群的资源。

开启 Fair Scheduler

设置yarn.resourcemanager.scheduler.class为 org.apache.hadoop.yarn.server.resourcemanager.scheduler.fair.FairScheduler（在 yarn-site.xml），如果你使用的是 CDH，那默认就是 Fair Scheduler（事实上，CDH 也不支持 Capacity Scheduler）。

队列设置

Fair Scheduler 通过 fair-scheduler.xml 文件来进行各种设置，这个文件的位置可以通过 yarn.scheduler.fair.allocation.file 属性来控制（在 yarn-site.xml）。如果没有这个文件，Fair Scheduler 采取的策略将是：每个任务都放在以当前用户命名的队列中，如果这个队列不存在，将会自动创建。

Fair Scheduler 也支持显式定义队列，就像 Capacity Scheduler 那样，下面是示例文件：

```
< ?xml version="1.0"?>
<allocations>
    <defaultqueueschedulingpolicy>fair</defaultqueueschedulingpolicy>
<queue name="prod">
    <weight>40</weight>
    <schedulingpolicy>fifo</schedulingpolicy>
</queue>
<queue name="dev">
    <weight>60</weight>
</queue>
<queue name="eng"></queue>
<queue name="science"></queue>

<queueplacementpolicy>
    <rule name="specified" create="false"></rule>
    <rule name="primaryGroup" create="false"></rule>
    <rule name="default" queue="dev.eng"></rule>
</queueplacementpolicy>
</allocations>
```

这里自定义了两个队列：prod 和 dev，权重比是 40:60，也就是说不采用均分的策略。每个队列可以有不同的调度策略，默认都是 fair，此外还有

FIFO、 Dominant Resource Fairness, 详细的配置信息可以查看官方文档。