Yarn 资源调度

yarn 集群的监控管理界面：

http://192.168.52.100:8088/cluster

jobHistoryServer 查看界面：

http://192.168.52.100:19888/jobhistory

yarn 的介绍：

yarn 是 hadoop 集群当中的资源管理系统模块，从 hadoop2.x 开始引入 yarn 来进行管理集群当中的资源（主要是服务器的各种硬件资源，包括 CPU，内存，磁盘，网络 IO 等）以及运行在 yarn 上面的各种任务。

总结一句话就是说：yarn 主要就是为了调度资源，管理任务等

其调度分为两个层级来说：

一级调度管理：

计算资源管理(CPU,内存，网络 IO，磁盘)　　硬件的资源

App 生命周期管理（每一个应用执行的情况,都需要汇报给 ResourceManager）

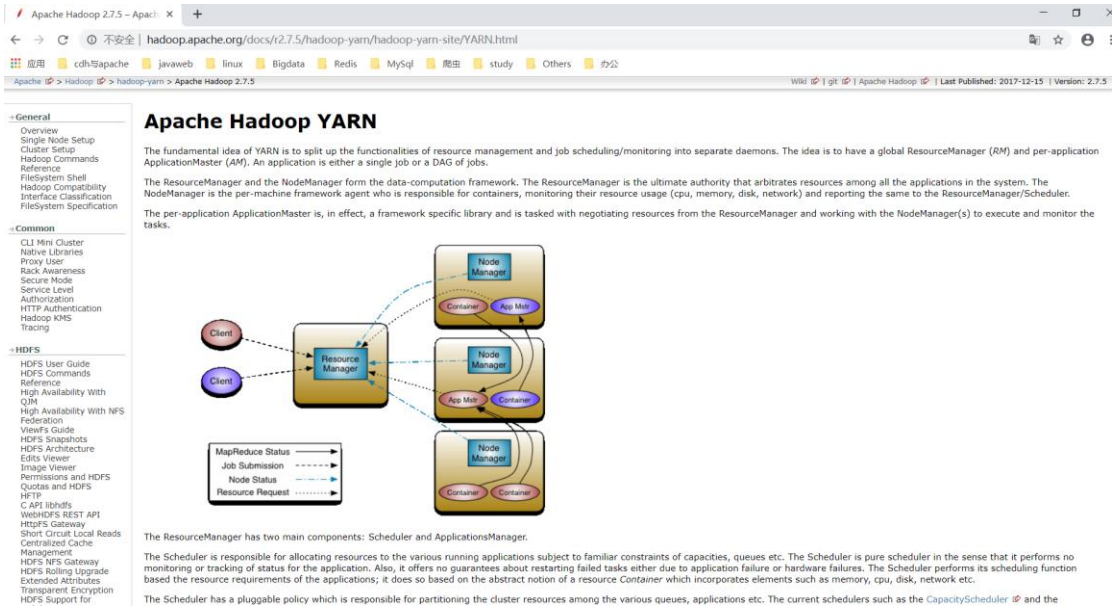
二级调度管理：　job 任务

App 内部的计算模型管理　(AppMaster 的任务精细化管理)

多样化的计算模型

yarn 的官网文档说明：

http://hadoop.apache.org/docs/r2.7.5/hadoop-yarn/hadoop-yarn-site/YARN.html



Yarn 的主要组件介绍与作用

yarn 当中的各个主要组件的介绍

ResourceManager：yarn 集群的主节点，主要用于接收客户端提交的任务，并对任务进行分配。

NodeManager：yarn 集群的从节点，主要用于任务的计算

ApplicationMaster：当有新的任务提交到 ResourceManager 的时候，ResourceManager 会在某个从节点 nodeManager 上面启动一个 ApplicationMaster 进程，负责这个任务执行的资源的分配，任务的生命周期的监控等

Container：资源的分配单位，ApplicationMaster 启动之后，与 ResourceManager 进行通信，向 ResourceManager 提出资源申请的请求，然后 ResourceManager 将资源分配给 ApplicationMaster，这些资源的表示，就是一个个的 container

JobHistoryServer：这是 yarn 提供的一个查看已经完成的任务的历史日志记录的服务，我们可以启动 jobHistoryServer 来观察已经完成的任务的所有详细日志信息

TimeLineServer：hadoop2.4.0 以后出现的新特性，主要是为了监控所有运行在 yarn 平台上面的所有任务（例如 MR，Storm，Spark，HBase 等等）

yarn 的发展历程以及详细介绍：

https://www.ibm.com/developerworks/cn/opensource/os-cn-hadoop-yarn/

yarn 当中各个主要组件的作用：

resourceManager 主要作用：

处理客户端请求

启动/监控 ApplicationMaster

监控 NodeManager

资源分配与调度

NodeManager 主要作用：

单个节点上的资源管理和任务管理

接收并处理来自 resourceManager 的命令

接收并处理来自 ApplicationMaster 的命令

管理抽象容器 container

定时向 RM 汇报本节点资源使用情况和各个 container 的运行状态

ApplicationMaster 主要作用：

数据切分

为应用程序申请资源

任务监控与容错

负责协调来自 ResourceManager 的资源，开通 NodeManager 监视容的执行和资源使用（CPU,内存等的资源分配）

Container 主要作用：

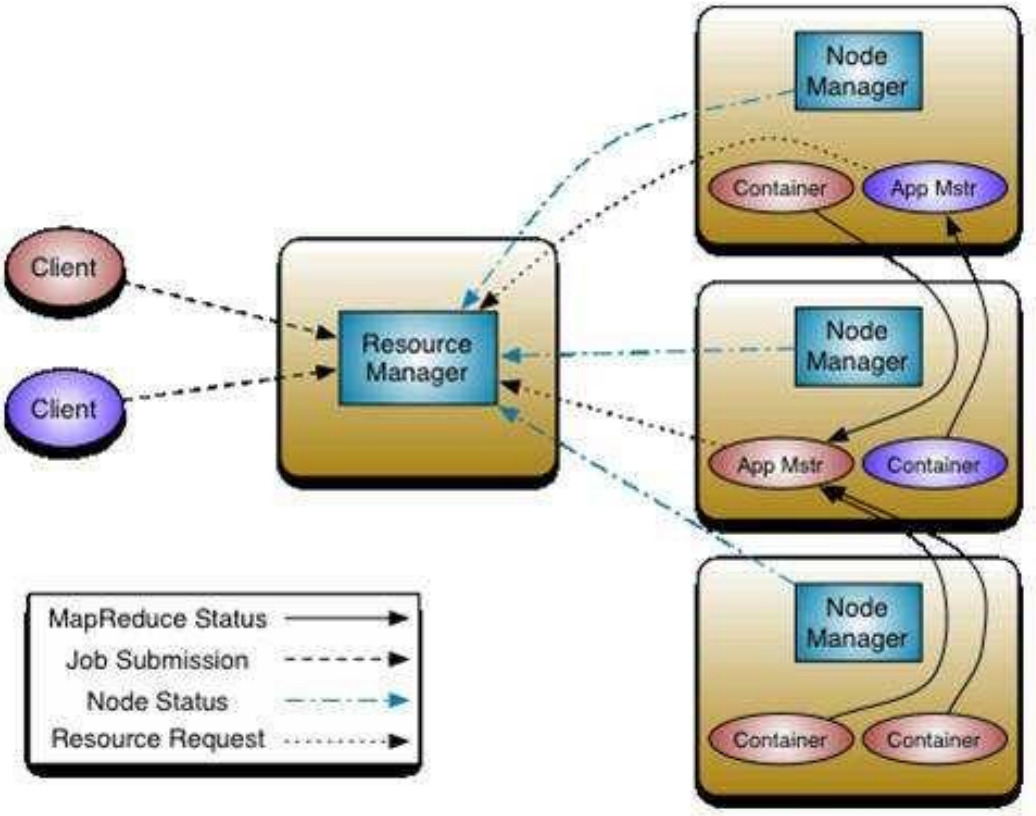
对任务运行环境的抽象

任务运行资源（节点，内存，cpu）

任务启动命令

任务运行环境

yarn 的架构



yarn 当中的调度器

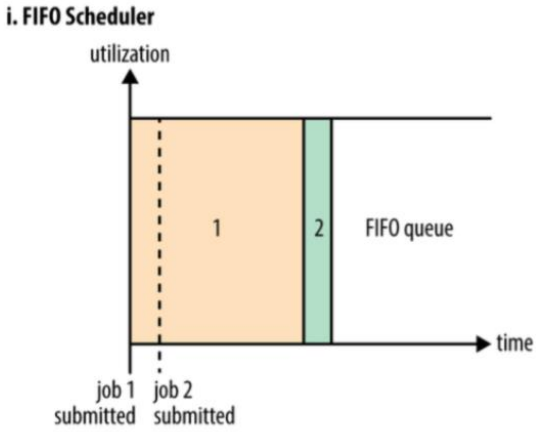
yarn 我们都知道主要是用于做资源调度，任务分配等功能的，那么在 hadoop 当中，究竟使用什么算法来进行任务调度就需要我们关注了，hadoop 支持好几种任务的调度方式，不同的场景需要使用不同的任务调度器

yarn 当中的调度器介绍：

第一种调度器：FIFO Scheduler （队列调度器）

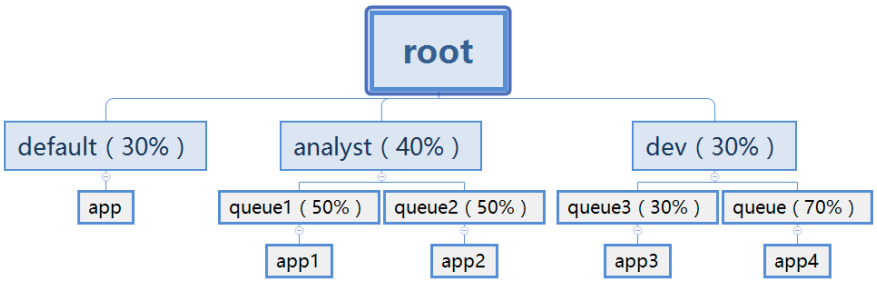
把应用按提交的顺序排成一个队列，这是一个先进先出队列，在进行资源分配的时候，先给队列中最头上的应用进行分配资源，待最头上的应用需求满足后再给下一个分配，以此类推。

FIFO Scheduler 是最简单也是最容易理解的调度器，也不需要任何配置，但它并不适用于共享集群。大的应用可能会占用所有集群资源，这就导致其它应用被阻塞。在共享集群中，更适合采用 Capacity Scheduler 或 Fair Scheduler，这两个调度器都允许大任务和小任务在提交的同时获得一定的系统资源。



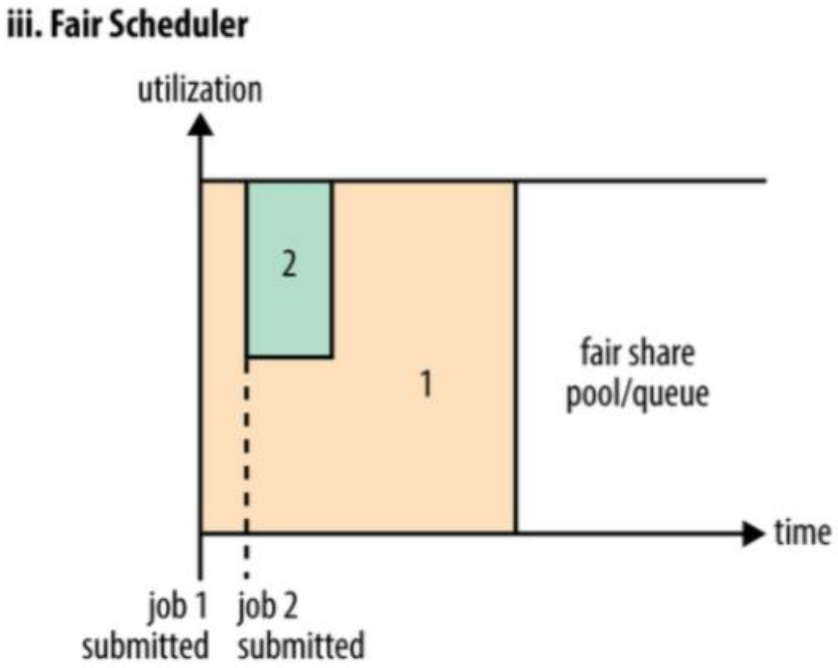
第二种调度器：capacity scheduler（容量调度器，apache 版本默认使用的调度器）

Capacity 调度器允许多个组织共享整个集群，每个组织可以获得集群的一部分计算能力。通过为每个组织分配专门的队列，然后再为每个队列分配一定的集群资源，这样整个集群就可以通过设置多个队列的方式给多个组织提供服务了。除此之外，队列内部又可以垂直划分，这样一个组织内部的多个成员就可以共享这个队列资源了，在一个队列内部，资源的调度是采用的是先进先出(FIFO)策略。



第三种调度器：Fair Scheduler（公平调度器，CDH 版本的 hadoop 默认使用的调度器）

Fair 调度器的设计目标是为所有的应用分配公平的资源（对公平的定义可以通过参数来设置）。公平调度在也可以在多个队列间工作。举个例子，假设有两个用户 A 和 B，他们分别拥有一个队列。当 A 启动一个 job 而 B 没有任务时，A 会获得全部集群资源；当 B 启动一个 job 后，A 的 job 会继续运行，不过一会儿之后两个任务会各自获得一半的集群资源。如果此时 B 再启动第二个 job 并且其它 job 还在运行，则它将会和 B 的第一个 job 共享 B 这个队列的资源，也就是 B 的两个 job 会用于四分之一的集群资源，而 A 的 job 仍然用于集群一半的资源，结果就是资源最终在两个用户之间平等的共享



使用哪种调度器取决于 yarn-site.xml 当中的

`yarn.resourcemanager.scheduler.class` 这个属性的配置

关于 yarn 常用参数设置

第一个参数：container 分配最小内存

yarn.scheduler.minimum-allocation-mb 1024 给应用程序 container 分配的最小内存

第二个参数：container 分配最大内存

yarn.scheduler.maximum-allocation-mb 8192 给应用程序 container 分配的最大内存

第三个参数：每个 container 的最小虚拟内核个数

yarn.scheduler.minimum-allocation-vcores 1 每个 container 默认给分配的最小的虚拟内核个数

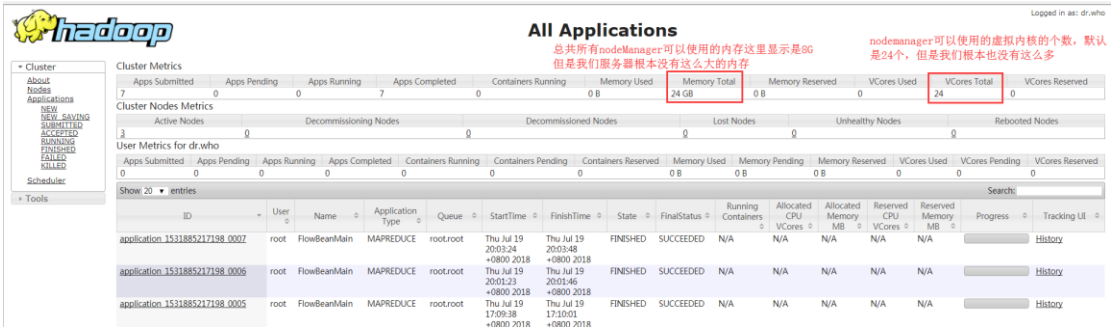
第四个参数：每个 container 的最大虚拟内核个数

yarn.scheduler.maximum-allocation-vcores 32 每个 container 可以分配的最大的虚拟内核的个数

第五个参数：nodeManager 可以分配的内存大小

yarn.nodemanager.resource.memory-mb 8192 nodemanager 可以分配的最大内存大小，默认 8192Mb

在我们浏览 yarn 的管理界面的时候会发现一个问题



我们可以在 yarn-site.xml 当中修改以下两个参数来改变默认值

定义每台机器的内存使用大小

yarn.nodemanager.resource.memory-mb	8192

定义每台机器的虚拟内核使用大小

yarn.nodemanager.resource.cpu-vcores	8

定义交换区空间可以使用的大小（交换区空间就是讲一块硬盘拿出来做内存使用）

这里指定的是 nodemanager 的 2.1 倍

yarn.nodemanager.vmem-pmem-ratio	2.1