Spark基本概念

Spark应用在集群中以独立的进程集的方式执行，由应用主程序(Driver)中的SparkContext对象来进行协同。SparkContext可以与多种Cluster Manager进行交互，例如Spark own Standalone cluster Manager，Mesos或者YARN，将应用程序提交到Cluster Manager中执行。Cluster Manager的作用是为应用分配资源：



SparkContext与ClusterManager建立连接后，Spark通过Cluster Manager在节点上启动executors，Executor是应用的实际计算和数据存储的进程。然后SparkContext将应用代码，一般定义成JAR或者python文件上传到Executors中，最后，SparkContext在executors中执行tasks。上图的执行框架有以下几个部分的注意点：

1. 每个应用程序有自己的executor进程，该进程持续整个应用周期，在executor的线程中运行tasks。这种方式使调度（Driver调度任务）端及执行端（在不同的JVM中执行的tasks）都能够使用资源隔离。在另一方面，如果不利用外部存储系统，数据不同在不同的应用之间共享。
2. Spark对Cluster Manager是透明的，其仅向CM申请资源来执行executor，这些executor启动后相互之间进行交互。这带来的好处是可以使用多种Cluster Manager。
3. Driver程序，在整个生命周期中都会接受从executor的网络连接。因此，driver programe和executor之间的网络必须相通。
4. Driver向集群中调度任务，因此要尽量靠近worker node相近，最好是同一个网段。如果从远程访问cluster，最好是使用RPC连接。

# 术语，基本概念

1. Application

用户在Spark上构建的程序，包含了Driver程序以及在集群中运行的程序代码

1. Application jar

用户应用程序的jar包，在一些场景用户将应用和依赖打到同一个b包中，但是用户不应该将hadoop和spark的jar包放到应用jar中，这些jar包应该在运行时加载

1. Driver

运行Application的main函数并创建SparkContext，SC的目的是为了准备Spark应用程序的运行环境。SC负责资源的申请、任务分配和监控等。当Executor运行结束后，Driver负责关闭SparContext

1. Cluster Manager

集群的资源管理器，在集群上获取资源的外部服务。以YARN为例，客户端程序向YARN申请需要的Memory,CPU等。然后CM会通过调度告诉客户端资源可以使用，然后客户端将程序送到每个Worker Node上面去执行。

1. Deploy mode

Driver进程的执行方式，在Cluster Mode下，Framework在集群中运行Driver。在客户端模式下，submitter在集群外执行driver。

1. Worker Node

集群中任何一个可以运行Spark应用代码的节点

1. Executor

在Worker Node上为某个应用启动的一个进程，该进程负责执行任务，并且负责将数据存到内存或者磁盘上，每个任务都有各自独立的Executor。Executor是一个执行Task的容器，主要执行包括：

初始化程序要执行的上下文SparkEnv，解决应用程序需要运行时jar包的依赖及加载类

同时还有一个ExecutorBachend像cluster manager汇报当前的任务状态，类似于hadoop的tasktracker和task。总的来说Executor是一个应用程序运行的监控和执行容器。

1. Jobs

执行任务，包含很多Task的并行计算，可以认为是Spark RDD中的action，每个action触发一个Job。用户提交的Job会提交给DAGScheduler，Job会被分解成Stage，Stage会被细化成Task，Task简单的说是在一个数据partition上单个数据处理流程。

1. Stage

每个Job被拆分为多组Task，每组任务被称为一个Stage就像Map Stage及Reduce Stage。

1. Task

被送到Executor上的工作单元

1. Partition

类似于Hadoop的Split，计算是以partition为单位进行。

# Submitting Applications

应用使用Spark-submit向任意类型的集群提交应用。以YARN为例，首先将HADOOP\_CONF\_DIR和YARN\_CONF\_DIR指向Hadoop配置文件，这些配置用户连接HDFS和YARN RM，配置文件为spark-env.xml，向YARN中提交程序的命令如下所示：

*$ ./bin/spark-submit --class path.to.your.Class --master yarn --deploy-mode cluster [options] <app jar> [app options]*

例如，向YARN中提交一个SparkPi程序，命令如下所示：

*./bin/spark-submit --class org.apache.spark.examples.SparkPi*

*--master yarn*

*--deploy-mode cluster*

*--driver-memory 2g --executor-memory 2g --executor-cores 1*

*--queue root.default*

*…/jars/spark-examples\_2.11-2.1.1.jar*

执行日志如下：

*client.RMProxy: Connecting to ResourceManager at fys2.cmss.com/10.139.9.124:8050*

*yarn.Client: Requesting a new application from cluster with 3 NodeManagers*

*yarn.Client: Verifying our application has not requested more than the maximum memory capability of the cluster (5120 MB per container)*

*yarn.Client: Will allocate AM container, with 2432 MB memory including 384 MB overhead*

*yarn.Client: Setting up container launch context for our AM*

*yarn.Client: Setting up the launch environment for our AM container*

*yarn.Client: Preparing resources for our AM container*

*yarn.Client: Application report for application\_1498704223336\_0006 (state: RUNNING)*

*yarn.Client: Application report for application\_1498704223336\_0006 (state: RUNNING)*

*yarn.Client: Application report for application\_1498704223336\_0006 (state: FINISHED)*

*17/07/07 22:06:44 INFO yarn.Client:*

*client token: N/A*

*diagnostics: N/A*

*ApplicationMaster host: 10.139.9.124*

*ApplicationMaster RPC port: 0*

*queue: root.default*

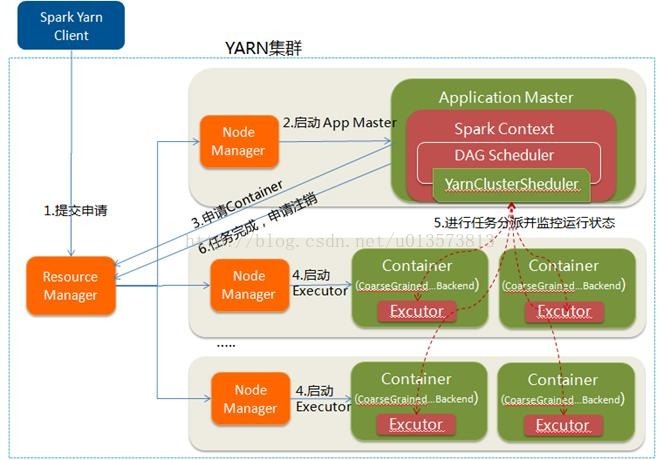
*start time: 1499436311936*

*final status: SUCCEEDED*

*tracking URL: http://fys2.cmss.com:8088/proxy/application\_1498704223336\_0006/A*

*user: spark*

Spark on yarn在Cluster Mode下的执行过程如下所示：



1. Spark客户端向ResourceManager提交程序
2. RM接收到请求后在集群中选择一个NodeManager分配Container，并在container中启动ApplicationMaster进程
3. 在AM进程中启动SparkContext
4. ApplicationMaster向RM申请到Container后，通知NM在container中启动Executor进程
5. SparkContext分配Task给executor,executor发送运行状态给ApplicationMaster

在上例中，各节点的进程如下所示：

*[yarn@fys3 ~]$ /usr/jdk64/jdk1.7.0\_67/bin*

*25826 ApplicationMaster*

*[yarn@fys2 ~]$ /usr/jdk64/jdk1.7.0\_67/bin*

*5898 CoarseGrainedExecutorBackend*

*5833 ContainerLocalizer*

*[yarn@fys1 ~]$ /usr/jdk64/jdk1.7.0\_67/bin*

*27184 CoarseGrainedExecutorBackend*