Spark standalone 任务调度流程

./bin/spark-submit --class example.Test \
--master spark://192.168.1.122:7077

SparkSubmit 进程调用 example.Test 的 main() 函数开始执行 app

example.Test.scala

```
def main() {
  val sc = new SparkContext()
  val rdd = sc.makeRDD()
  val finalRDD = rdd.transformation()
  val result = finalRDD.action()
}
```

SparkContext

action() 调用 SparkContext.runJob()　调用

DAGScheduler

dagScheduler.runJob()

根据 finalRDD 得到 finalStage

从后向前扫描整个 RDD DAG 图，遇到 ShuffleDependency 就断开 DAG 图，生成一个stage

// 打印一些 stage 信息
logInfo("Got job * with * output partitions")
logInfo("Final stage: stage + (stageName)")
logInfo("Parents of final stage: stage.parents")
logInfo("Missing parents: missingParentStages")

将 jobStart 的消息 post 到 ListenerBus 上

提交最后一个 finalStage

如果 finalStage 有 parentStages，先提交 parentStages，依此类推

如果当前 stage 已经是第一个 stage，那么开始提交该 stage 里面的 tasks

提交 tasks 的逻辑

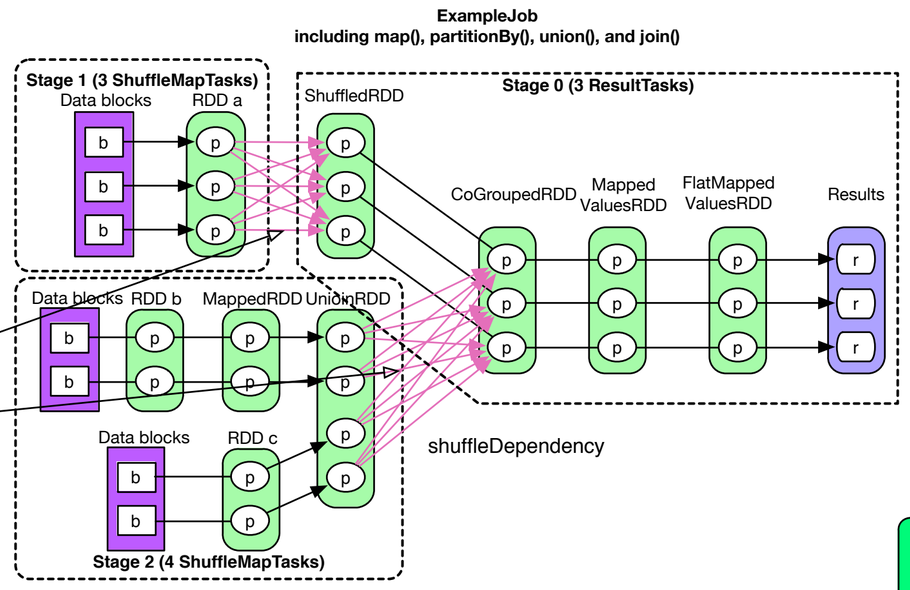
将 stage 信息（shuffleMapStage 还是 ResultStage）序列化并广播到各个 workers

生成该 stage 中的 tasks，用 TaskSet 表示
val tasks = Seq[ShuffleMapTask/ReduceTask]

向 taskScheduler 提交该 stage 中的 tasks
taskScheduler.**submitTasks**(new TaskSet(tasks.ToArray, stage.id))

当前 stage 已经完成
markStageAsFinished(stage, None)

依次提交其他未运行的 stages



taskScheduler: TaskSchedulerImpl

提交当前 stage 中 tasks (也就是 taskSet)的逻辑
submitTasks(taskSet: TaskSet)

为 taskSet 分配一个 taskSetManager

将 taskSetManager 放到 taskSet 调度器 （FIFO 或 FairSchedulableBuilder）里面等待调度

让 SparkDeploySchedulerBackend 执行提交逻辑

确定 tasks 要分配到哪些 executor 上（round-robin 方式）

resourceOffers(offers: Seq[WorkerOffer])

将目前可用的 workOffers (也就是可用的 executors) 随机打乱

用于保存每个 executor 上应该运行的 tasks
val tasks = shuffledOffers.map(offer => Array[TaskDescription](offer.cores))

从 FIFO/FairSchedulableBuilder 的 schedululableQueue 中提取出已经排好序的 taskSets（可以并行执行的 stages）

对于每一个 taskSet，为其分配资源
resourceOfferSingeTaskSet(taskSet, maxLocality, tasks)

return tasks

// tasks: [to-be-allocated tasks in executor 1, to-be-allocated tasks in executor 2, …]
resourceOfferSingeTaskSet(taskSet, maxLocality, tasks: Seq[Array[TaskDescription]])

对于每一个可用的 executor，如果 taskSet 中的 tasks 可以放到这个 executor 上，那么就在这个 executor 上分配 tasks

schedulableBuilder

如果一个 stage 有多个可以并行执行的 parent stages，会将这些 stages 对应的 taskSets 都放在一起进行 FIFO 或者 Fair 调度

schedulableQueue: ConcurrentListedQueue[TaskSetManager]

FIFOSchedulableBuilder

addTaskSetManager(manager)

FairSchedulableBuilder

addTaskSetManager(manager)

backend: SparkDeploySchedulerBackend

确定目前可用的 executors 及相应的 freeCores

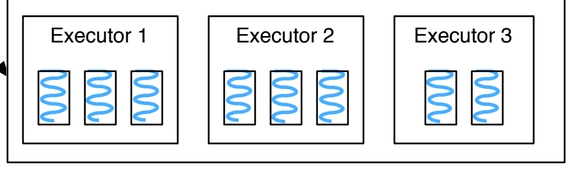
val workOffers = seq[(activeExecutorId, executorHost, executor.freeCores)]

确定每个 executor 上应该运行哪些 tasks (由 resourceOffers() 完成)，并将 tasks 提交给相应的 executor 执行

tasks = scheduler.resourceOffers(workOffers)
launchTasks(tasks)

对每个 task 将其序列化后提交给相应的 executor 执行

tasks: Seq[ArrayBuffer[TaskDescription]]



TaskSetManager

// 将 task 及其依赖的 files/jars 进行序列化
val serializedTask = Task.serializeWithDependencies(task, addedFiles/Jars)

向 dagScheduler 发送 taskStarted 消息

返回 new TaskDescription(taskId, attemptNumber, execId, serialized task)