wordcount 执行流程

Spark 会在有数据的机器上创建分区，与 NameNode 交互，知道数据在哪些机器上，并在那些机器上创建分区。

Spark 启动 Executor 时并不知道数据在哪些机器上。Driver 与 Master 建立连接后，Master 分配资源并启动 Executor。

1. wordcount 执行流程

（1）sc 是与 spark 集群交互的入口。

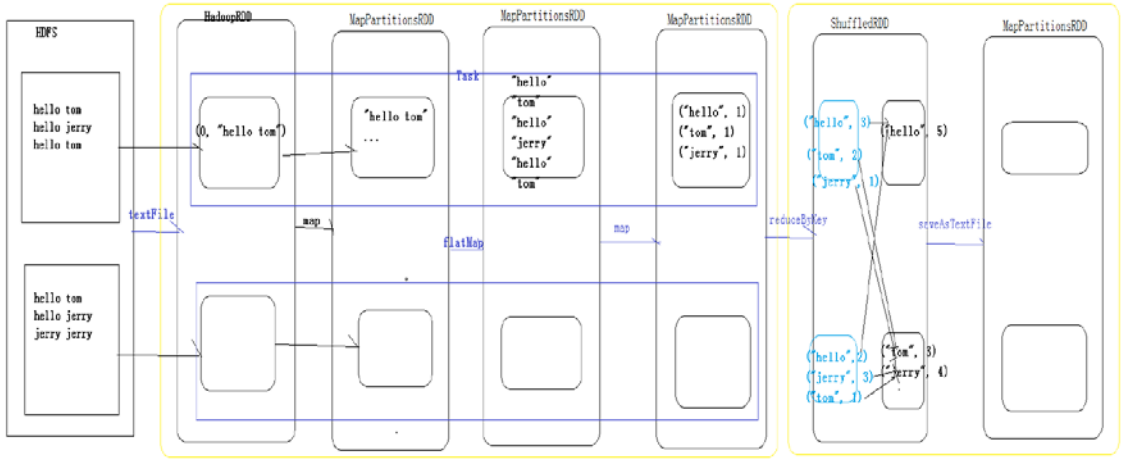
（2）textFile()从外部存储介质读取数据产生两个 RDD：HadoopRDD 读取 KV 数据，MapPartitionsRDD 对每个分区进行 map 操作。可以指定分区数（默认为 2）。若从 hdfs 读取数据，不指定分区数，则默认两个分区，且一个数据块对应一个分区。

（3）flatMap()产生 MapPartitionsRDD，针对每个分区调用 map()方法。

（4）map()产生 MapPartitionsRDD。

（5）reduceByKey()先对每个分区中的数据分组聚合，再将所有分区数据聚合；可能会产生 shuffle，shuffle 时产生 ShuffledRDD；若数据已经分组，且分区数不变化，则不产生 shuffle。

（6）saveAsTextFile()产生 MapPartitionsRDD。



注：

1) 任务在触发 Action 执行时使用递归算法从后往前推断，遇到宽依赖（Shuffle）则切分 Stage；
2) 递归的终止条件为当前 RDD 没有父 RDD；
3) 分区在创建 RDD 时创建完成，stage 提交后分区从对应的 block 中读取 KV 数据；
4) Spark 在计算时边读数据边处理计算，数据默认放入内存，若内存不足，则落入磁盘；
5) Shuffle 过程包含 3 个阶段：对每个分区数据进行局部聚合；从上游的分区中拉取数据；进行全局聚合；shuffle 时若内存能放下，也需要落入磁盘。出现问题要恢复，代价高。
6) 分区数对应了提交任务的 task 数，每个 task 中都是流水线作业；
7) 为什么 mapreduce 计算慢？不停的计算并落入磁盘，mapreduce 中转换操作是有限的；而 spark 所有转换都在一个 task 流水线中，所以计算快。

补充：

（1）RDD 的依赖关系是如何保存的？

新生成的 RDD 保存上游 RDD 的引用（this），即可进行推断。

（2）Spark 为什么要划分 Stage 来计算？

划分 stage 的依据是宽依赖（是否存在 shuffle）。计算时将任务划分成多个 stage，前一个 stage 计算完成再计算下一个 stage 任务。